GRAFICA
E SUONO
per il C64 / C128
C64 Personal
Computer
HOME E PERSONAL COMPUTER
GEOS
comandi e
applicazioni
CON
FLOPPY DISK
GRUPPO EDITORIALE
JACKSON

GRAFICA E SUONO

per il C64/C128 C64 Personal Computer

GRUPPO
EDITORIALE
JACKSON
Via Rosellini, 12
20124 Milano

INDICE

Introduzione

Il Commodore 64 può produrre effetti sonori e grafica fra i migliori ottenibili con un home computer. Alcune di queste gradevoli caratteristiche, tuttavia, possono risultare piuttosto complicate da padroneggiare. Anche se non siete degli esperti programmatori, *C64: suono e grafica* vi aiuterà ad apprendere quelle tecniche che vi permetteranno di apprezzare appieno il vostro calcolatore.

Sia i principianti che i programmatori più esperti troveranno parecchi suggerimenti interessanti e di immediato utilizzo. Tutti i programmi sono pronti per essere copiati e mandati in esecuzione.

Fate attenzione, in particolare, a «Blocco per schizzi ad alta risoluzione». Potete utilizzare questo programma anche se non avete mai avuto un calcolatore prima d'ora. «Blocco per schizzi ad alta risoluzione» vi consente di tracciare disegni a due e a quattro colori e di registrarli su disco o nastro, in modo da poterli vedere ogni volta che lo desiderate.

Completa il testo una parte dedicata ai comandi e alle applicazioni del sistema operativo GEOS.

PRIMA PARTE

Capitolo uno

Primi passi

Capitolo uno

Un computer grafico

Gregg Keizer

«Un computer grafico» vi presenta gli aspetti fondamentali della gestione della grafica del Commodore 64.

Avete tolto il vostro Commodore 64 dal suo imballaggio, lo avete collegato alla televisione, controllato i collegamenti ed aperto il *Manuale d'uso.* Pensate di essere pronti per programmare. Ma per qualche strana ragione provate un senso di vuoto allo stomaco, un momento di ansia alla prospettiva di affrontare il calcolatore. Quando accendete il calcolatore appare il messaggio READY ed il cursore lampeggia. E ora? Ricordandovi delle complicate dimostrazioni che il venditore vi ha mostrato, degli affascinanti giochi che hanno cercato di vendervi, decidete di mettere alla prova le vostre capacità con un programma di grafica. Dopo tutto, il Commodore 64 dispone di una grafica a colori evoluta. È stampato sulla scatola, molto chiaro, ed il negoziante vi ha ripetuto più di una volta che questo calcolatore ha delle capacità grafiche impressionanti. Un blocco per schizzi piuttosto costoso, pensate. Però sarete in grado di tracciare disegni e creare scenari altrettanto espressivi di quelli delle dimostrazioni e dei giochi che vi hanno mostrato.

Leggete da capo a fondo il *Manuale d'uso* e ben presto vi rendete conto che non sarà così semplice. Sembra che anche le immagini più elementari richiedano righe su righe di numeri e simboli indecifrabili. Vi si dice che dovrete apprendere un nuovo linguaggio, il BASIC, prima di poter creare quegli effetti grafici con cui la vostra fantasia stava, ben presto, riempiendo lo schermo vuoto. Iniziano i primi momenti di frustrazione, quindi impazienza, poi un senso di disperazione. Siete convinti che non sarete mai in grado di far fare al calcolatore ciò che volete.

Sbagliato. C'è una speranza. Una volta che avrete superato lo shock iniziale di dover imparare cose nuove per fare cose nuove, vi

accorgerete che non è poi così difficile. Effetti grafici come quelli di *«Torneo»* o *«Donkey Kong Jr.»* possono richiedere un po' di tempo per essere realizzati, ma i principi basilari della grafica su computer sono piuttosto semplici da afferrare, se avete qualche esperienza della programmazione in linguaggio BASIC. Non è nemmeno necessaria una laurea in matematica o in informatica. Siete già in possesso delle caratteristiche necessarie per scrivere programmi di grafica. Desiderio di apprendere, sperimentare e creatività. Se avete tutto ciò, vi divertirete, ben presto, a creare grafici con il C64.

La grafica del Commodore 64

Il Commodore 64 *è* effettivamente un calcolatore grafico. *Ha* realmente capacità grafiche *stupefacenti.* Tutto ciò di cui ha bisogno è qualcuno che gli dica che cosa fare. Questo qualcuno siete voi, il programmatore. Anche se voi non siete in grado di pensare a voi stessi come a dei programmatori (l'immagine che vi balza alla mente è quella di qualcuno seduto davanti a una tastiera alle tre del mattino), ciò è quello che siete. Senza un programmatore il vostro computer continuerebbe a far lampeggiare il suo cursore, attendendo pazientemente un comando. Se ne starebbe lì fermo per sempre, senza fare niente, a meno che voi lo *programmiate* per fare qualcosa.

Quando voi programmate una istruzione nel vostro calcolatore, *allora* esso comincia a lavorare. E le sue capacità grafiche verranno messe a dura prova da voi.

Il Commodore 64 utilizza il suo Chip di Interfaccia Video 6567 (noto come chip VIC-II) per creare gli effetti grafici che voi programmerete. Esiste tutta una serie di formati grafici prodotti da questo chip, inclusi uno schermo in formato testo di 25 righe da 40 colonne, uno schermo ad alta risoluzione di 320x200 punti e sprite, i piccoli oggetti mobili che potete disegnare e utilizzare nei giochi. Non solo il C64 ha diversi formati grafici, ma questi formati possono anche essere mescolati. Potete combinare il formato testo con il formato ad alta risoluzione per creare un disegno ricco di dettagli nella parte superiore dello schermo e delle scritte nella parte inferiore. Gli sprite possono essere mescolati con qualsiasi altro formato, rendendo più semplice la programmazione dei giochi.

Il formato grafico più semplice, e quello con cui senz'altro voi comincerete, è il formato testo. Non lasciatevi ingannare dal nome; potete creare figure grafiche complesse ed espressive sullo scher-

mo anche in formato testo. Contrariamente ad altri calcolatori, non è necessario che voi poniate il C64 in questo formato tramite un comando esplicito. Nel momento in cui accendete il vostro calcolatore vi trovate automaticamente in questo formato. La creazione di grafici in alta risoluzione e l'uso degli sprite sono più difficili e richiedono più tempo, benché la ricompensa possa essere maggiore. Questi formati sono completamente coperti dagli ultimi capitoli di questo libro, ma dal momento che voi state appena iniziando con la grafica ci limiteremo, tanto per cominciare, ad occuparci del formato testo.

Come si colora il testo

Quando accendete il Commodore 64 il video è in una combinazione monocolore. Un testo in azzurro appare su uno sfondo blu, con una cornice azzurra. Questo è il colore per difetto, vale a dire il colore che il calcolatore utilizzerà, a meno che voi gli ordiniate di fare altrimenti.

Cambiare il colore del testo è piuttosto semplice. In effetti, esiste più di un modo per farlo. Il modo più semplice, e voi dovreste già conoscerlo, consiste nell'usare i tasti colore della tastiera.

Il C64 ha sedici colori di testo con cui voi potete lavorare. I primi otto sono disponibili usando il tasto CTRL e un tasto numerico, mentre il secondo gruppo di otto è disponibile usando il tasto Commodore anch'esso accoppiato ad un tasto numerico. La tavola 1 mostra i colori utilizzabili per il testo e le combinazioni di tasti richieste.

Tavola 1. Colori del Commodore 64

Colore	**Tasti**
NERO	CONTROL 1
BIANCO	CONTROL 2
ROSSO	CONTROL 3
BLU VERDE	CONTROL 4
PORPORA	CONTROL 5
VERDE	CONTROL 6
BLU	CONTROL 7
GIALLO	CONTROL 8
ARANCIO	COMMODORE 1
MARRONE	COMMODORE 2

ROSSO CHIARO	COMMODORE 3
GRIGIO 1	COMMODORE 4
GRIGIO 2	COMMODORE 5
VERDE CHIARO	COMMODORE 6
AZZURRO	COMMODORE 7
GRIGIO 3	COMMODORE 8

Premendo CTRL e 1, ad esempio, il cursore diventa nero. Qualsiasi testo che voi battiate d'ora in avanti apparirà in nero. Ma se premete RETURN, compare un messaggio d'errore. Sullo schermo si può leggere SYNTAX ERROR. Che cosa sta succedendo?

Per cambiare il colore del testo dovete utilizzare un comando PRINT. Solo in questo caso il vostro calcolatore capirà che gli chiedete di cambiare il colore del testo. Fintantoché usate il comando PRINT ed includete le istruzioni tra i doppi apici ("), il C64 seguirà le vostre direttive.

```
PRINT "[BLK]STO SCRIVENDO IN NERO"
```

Se premete CTRL e 1, potete notare la comparsa di un simbolo in negativo sullo schermo. Questo è il simbolo che il computer usa per ricordarsi quale colore deve essere mostrato. Non preoccupatevi di ricordare quale simbolo si accoppia a ciascun colore; il computer lo fa al vostro posto.

La riga che voi avete copiato rimarrà nel colore standard azzurro, fino a che non premerete RETURN. Solo dopo che lo avrete fatto il cursore, e qualsiasi testo che avrete battuto in seguito, apparirà nero. Ciò che avete appena fatto è stato dire al calcolatore di cambiare il colore di testo. A meno che venga cambiato nuovamente, il C64 continuerà ad usare questo colore.

Per ritornare al colore originale si può sia premere contemporaneamente i tasti RUN/STOP e RESTORE che battere

```
PRINT"[(7)]"
```

e premere RETURN. Ora il testo appare nuovamente in azzurro.

Qualsiasi carattere che può essere battuto da tastiera, inclusi i caratteri grafici standard che appaiono quando i tasti SHIFT o Commodore vengono utilizzati contemporaneamente ad un altro tasto, può essere mostrato in diversi colori. Ma probabilmente vi sarete accorti dell'esistenza di un problema. Quando premete RETURN il testo all'interno dei doppi apici viene stampato, ma non potete mostrarlo nuovamente senza ribattere l'intera riga.

Quindi, benché abbiate comandato al calcolatore di compiere una operazione, in realtà non l'avete programmato. Senza ulteriori istruzioni il C64 eseguirà il vostro comando una sola volta, quindi se ne dimenticherà. Per far sì che se ne ricordi, le istruzioni devono essere scritte sotto forma di programmi. In altre parole, si devono assegnare i numeri di riga e bisogna comandare al calcolatore di eseguire il programma tramite una RUN. Il cambiamento è minimo ed ha il seguente aspetto:

```
10 PRINT"[BLK]STO SCRIVENDO IN NERO"
```

È un programma di una sola riga, ma funzionerà all'infinito finché continuerete a battere RUN ogni volta. L'effetto è sempre lo stesso, poiché il testo verrà continuamente riproposto in nero, a meno che si effettuino modifiche o si usi RUN/STOP-RESTORE. Ma il calcolatore continuerà a ricordarsi i comandi. In effetti, non li dimenticherà finché voi, il programmatore, non gli direte di farlo o finché il calcolatore non verrà spento.

Il breve programma che segue dimostra come i colori possano essere cambiati quante volte si desidera e che il colore permane finché non viene nuovamente cambiato. Notate come, alle righe da 70 a 100, il tasto Commodore venga utilizzato per scegliere i colori appartenenti al secondo gruppo di otto.

Programma 1. Cambiamenti di testo

```
 10 PRINT"[BLK]UNA DIMOSTRAZIONE"
 20 PRINT"[WHT]DEI COLORI"
 30 PRINT"[RED]CHE SONO DISPONIBILI"
 40 PRINT"[CYN]SUL COMMODORE 64"
 50 PRINT"[PUR]E' FACILMENTE "
 60 PRINT"[GRN]REALIZZABILE"
 70 PRINT"[(1)]RITORNARE"
 80 PRINT"[(2)]AL COLORE"
 90 PRINT"[(3)]ORIGINALE NON E' POI"
100 PRINT"[(7)]COSI' DIFFICILE"
110 GOTO 110
```

Benché ogni riga stampi in colori differenti, il testo sarebbe rimasto in rosso chiaro (il colore selezionato in riga 90), se non fosse stato riportato al colore standard azzurro in riga 100.

Capitolo uno

Caratteri grafici a colori

I caratteri grafici possono essere mostrati in vari colori, nello stesso modo in cui lo si fa per il testo. Questi caratteri sono quelli mostrati sulla faccia dei tasti rivolta verso l'operatore e che compaiono premendo i tasti SHIFT o Commodore, quindi il tasto voluto. Premendo SHIFT ed un tasto si ottiene il simbolo che compare a destra della faccia, mentre usando il tasto Commodore accoppiato ad un tasto si ottiene il simbolo di sinistra.

Questi caratteri grafici fanno parte del set standard Commodore e fanno del C64 un potente strumento grafico. Non è necessario che vi disegniate dei caratteri personalizzati, se non lo desiderate, inoltre ne avete a disposizione più di quelli offerti dalla maggior parte dei calcolatori. Usandoli, potete tracciare disegni, creare caratteri per giochi ed inventare nuove figure. Parecchi giochi, ad esempio, sono creati sul C64 solo utilizzando i caratteri grafici standard. Nel disegnare e creare i vostri disegni sullo schermo, li userete più di qualsiasi altro carattere.

Un blocco per schizzi

Il Commodore 64 può non essere altrettanto semplice da usare di una tavoletta grafica (un blocco per schizzi elettronico), ma può rivestire lo stesso ruolo, e per di più a colori. Pensate allo schermo come a un foglio di carta da lucido largo 40 colonne e alto 25 righe. In effetti, avere un foglio di carta da lucido su cui tracciare questa quadrettatura sarebbe di grande aiuto.

Ogni quadretto della carta da lucido rappresenta un carattere dello schermo del C64. Potete riempire ciascun quadretto con qualsiasi carattere della tastiera, dai caratteri alfanumerici a quelli grafici. Schizzare il vostro disegno su carta da lucido, decidere quali caratteri grafici utilizzare, e persino colorare il disegno con una matita, vi darà un'idea più chiara del risultato finale.

Potete trasformare lo schermo ed il calcolatore in un blocco per schizzi. Usando la barra spaziatrice, o i tasti cursore, potete piazzare caratteri alfanumerici o grafici in qualsiasi punto dello schermo. Quando arrivate al termine di una riga non premete RETURN; usate invece SHIFT-RETURN, che sposterà il cursore sulla riga successiva, senza mostrare la scritta READY. In questo modo potete muovervi per lo schermo a vostro piacimento, inserendo nuovi caratteri grafici, cancellandone altri, finché il disegno non vi soddisfa. Sapete esattamente quale sarà l'aspetto conclusivo del dise-

gno.

Così, come quando inserite una istruzione PRINT senza numero di riga, anche questo disegno andrà perduto una volta che siano stati premuti i tasti RUN/STOP-RESTORE. Per costringere il calcolatore a ricordarsi il vostro disegno, lo stesso dovrà essere scritto sotto forma di programma. Questo significa, sfortunatamente, duplicare il disegno che avete appena terminato, aggiungendo questa volta i numeri di riga, le istruzioni PRINT e i doppi apici. Il programma che segue è un esempio di uno schizzo completo, rappresentate una vista dall'alto di una tavola da biliardo con un giocatore pronto a colpire la palla.

Programma 2. Biliardo

```
10 PRINT"{CLR}"
20 PRINTTAB(16)"{BLK}{ 4 GIU'}O[< 7 U>]P"
30 PRINTTAB(16)"[<J>]{ 7 DES}[<L>]"
40 PRINTTAB(16)"[<J>]{ 7 DES}[<L>]"
50 PRINTTAB(16)"[<J>]{DES}{ 5 Q}{DES}[<L>]
   "
60 PRINTTAB(16)"[<J>]{ 2 DES}{ 3 Q}
   { 2 DES}[<L>]"
70 PRINTTAB(16)"[<J>]{ 3 DES}Q{ 3 DES}
   [<L>]"
80 PRINT
90 PRINTTAB(16)"[<J>]{ 7 DES}[<L>]"
100 PRINTTAB(16)"[<J>]{ 7 DES}[<L>]"
110 PRINTTAB(16)"[<J>]{ 2 DES}{WHT}{DES}W
    { 3 DES}{BLK}[<L>]"
120 PRINTTAB(16)"[<J>]{ 3 DES}G{ 3 DES}
    [<L>]"
130 PRINTTAB(16)"L[< 3 O>]G[< 3 O>]@"
140 PRINTTAB(16)"{ 4 DES}G"
150 PRINTTAB(16)"{ 4 DES}G"
160 PRINTTAB(16)"{ 4 DES}G[<2>]I"
170 PRINTTAB(16)"{ 5 DES}G"
180 PRINTTAB(16)"{ 3 DES}UEI"
190 PRINTTAB(16)"{ 3 DES}JFK"
200 GOTO200
```

L'istruzione TAB (16) utilizzata in ogni riga assicura che il disegno abbia il margine sinistro allineato. Il simbolo ■ rappresenta gli spostamenti del cursore verso destra, per creare spazi vuoti quando sono necessari. Notate come il colore venga cambiato in nero in riga 20; in bianco, quindi ancora in nero in riga 110 ed infine in marrone in riga 160. Anche se il disegno non sembra combaciare perfettamente nel programma mentre lo copiate, quando batterete RUN apparirà esattamente come lo volete. Le irregolarità sono dovute alla presenza dei comandi di controllo colore in alcune righe, oltre alla comparsa dei numeri di riga da tre cifre a metà del programma.

Realizzare dei disegni in questo modo può sembrare un'impresa piuttosto complicata, ma una volta che avrete copiato e registrato questo programma sarete in grado di mandarlo in esecuzione (per mezzo del comando RUN) tutte le volte che vorrete. Se registrerete il programma su nastro o disco, esso non andrà perduto quando toglierete la corrente o quando lo schermo verrà cancellato per un nuovo programma. Qualche esperimento con i vostri disegni vi mostrerà le capacità grafiche tramite l'istruzione PRINT del Commodore.

La funzione CHR$

Un altro modo per mostrare caratteri grafici, caratteri alfanumerici e colori sullo schermo, con il C64, consiste nell'usare la funzione CHR$ (si pronuncia «character string»), che fornisce un carattere basato su un numero di codice compreso tra 0 e 255. Ogni carattere e colore che il Commodore 64 è in grado di mostrare è codificato in questo modo. La maggior parte dei manuali, compreso il *Manuale d'uso del Commodore 64,* (che viene unito al vostro calcolatore), comprende una tavola dei valori della funzione CHR$ (vedi l'Appendice F). Per stampare un qualsiasi carattere tutto ciò che dovete fare è digitare:

```
PRINT CHR$(N)
```

dove N è un numero compreso tra 0 e 255. Ad esempio, provate a battere questa riga:

```
PRINT CHR$(65)
```

dovreste veder apparire il carattere A sullo schermo.

Se non avete a disposizione un manuale che comprenda i valori dei simboli di CHR$, potete trovarli da voi usando la funzione:

```
PRINT ASC("X")
```

dove X è un qualsiasi tasto premuto. Battete:

```
PRINT ASC("A")
```

dovreste veder apparire il valore di CHR$, 65, associato al carattere A. Questo sistema risulta molto pratico quando cercate i valori di CHR$ associati al secondo gruppo di otto colori. Parecchi manuali non comprendono i valori di CHR$ per questi colori, oppure non li elencano separatamente. Ad esempio, se battete:

```
PRINT ASC("[<2>]")
```

apparirà il numero 159, vale a dire il numero di CHR$ associato al marrone.

Usando CHR$, potete duplicare qualsiasi comando che possa essere battuto da tastiera. Ad esempio, una dimostrazione di testi in diversi colori avrà l'aspetto fornito dal programma seguente, usando la funzione CHR$ invece dei caratteri, battuti da tastiera, inclusi nei doppi apici:

Programma 3. Cambiamenti di testo con la funzione CHR$

```
5 PRINTCHR$(147)
10 PRINTCHR$(144)"UNA DIMOSTRAZIONE"
20 PRINTCHR$(5)"DEI COLORI"
30 PRINTCHR$(280)"CHE SONO DISPONIBILI"
40 PRINTCHR$(159)"SUL COMMODORE 64"
50 PRINTCHR$(156)"E' FACILMENTE"
60 PRINTCHR$(30)"REALIZZABILE."
70 PRINTCHR$(149)"RITORNARE"
80 PRINTCHR$(150)"AL COLORE"
90 PRINTCHR$(151)"ORIGINALE NON E' POI"
100 PRINTCHR$(154)"DIFFICILE"
110 GOTO 110
```

Questo programma è quasi identico al programma 1, tranne che per il fatto di utilizzare i valori di codice della funzione CHR$. La istruzione CHR$ (147) in riga 5 funge da istruzione di cancellazione dello schermo. Notate che, per le righe da 70 a 100, i valori di CHR$ corrispondono al secondo gruppo di otto colori, quelli che vengono normalmente mostrati quando si utilizza il tasto Commodore accoppiato ad un tasto numerico. La riga 100 fa tornare il colore all'azzurro standard e infine la riga 110 tiene il programma bloccato su di un ciclo chiuso, in modo che la scritta READY non rovini l'uscita video.

Confrontate i due metodi di cambiamento dei colori mostrati dai programmi 1 e 3. Inserire i valori dei codici di CHR$ richiede più tempo, più battute, ma produce il medesimo risultato. Per questa ragione l'uso della funzione CHR$ nella grafica è in parte limitativo. Spesso esistono modi più semplici per ottenere gli stessi risultati. Tuttavia, a volte troverete delle applicazioni in cui la funzione CHR$ è più utile, specialmente se state facendo esperimenti con rapidi cambiamenti di colore o di caratteri.

Se volete, ad esempio, riempire lo schermo di caratteri, oppure mostrarli in vari colori, la funzione CHR$ è molto efficace. Dal momento che potete dare come argomento della funzione CHR$ il valore di una variabile (a patto che questo valore sia compreso nei limiti prescritti), una dimostrazione di grafica casuale può essere più facilmente realizzata con il metodo suddetto. Il Programma 4 ne è un esempio.

Programma 4. CHR$ casuale

```
10 PRINT CHR$(147)
20 A=(191*(RND(9)))+34
30 IF A>129 AND A<149 THEN 20
40 PRINT CHR$(A);
50 GOTO20
```

Quando questo programma è in esecuzione riempie lo schermo di caratteri casuali, cambiandone contemporaneamente il colore. Ottiene questo risultato scegliendo in riga 20 un numero casuale compreso tra 34 e 191, che diventa la variabile A. In riga 40 viene stampato il carattere ricavato dalla funzione CHR$(A) ed il programma ricicla. Le uniche eccezioni sono rappresentate dai valori

della funzione CHR$ compresi tra 130 e 148. Non escludere questi valori fa compiere al programma strane cose, come potete vedere eliminando semplicemente la riga 30.

Ottenere gli stessi risultati con le semplici istruzioni PRINT richiederebbe molte più righe e più spazio nella memoria del vostro calcolatore e risulterebbe anche molto più lento.

Riempire lo schermo con caratteri e colori casuali può sembrare interessante, ma è piuttosto difficile trovargli una applicazione pratica. Qualcosa di più utile, e che utilizzi ancora la funzione CHR$, può assomigliare al seguente programma:

Programma 5. Scacchiera

```
10 CL=158
20 PRINT CHR$(147);CHR$(CL)
30 FOR A=1 TO 11
40 FOR X=1 TO 19
50 PRINT CHR$(18)" "CHR$(146)" ";
60 NEXT X:PRINT
70 FOR X=1 TO 19
80 PRINT CHR$(146)" "CHR$(18)" ";
90 NEXT X:PRINT
100 NEXT A
110 PRINT CHR$(154)
120 GOTO 120
```

Commenti al programma

Riga	**Commento**
10	La variabile CL contiene il codice corrispondente al colore usato nel programma. Cambiandolo, si altera il colore della scacchiera.
20	Si cancella lo schermo e si cambia il colore.
30	Se non avete mai usato un ciclo FOR-NEXT prima d'ora, vi potrà sembrare strano. Tutto ciò che fa è ripetere qualcosa; in questo caso le righe da 40 a 90 vengono ripetute 11 volte prima della fine del programma.
40	Questo ciclo fa sì che la riga seguente venga ripetuta 19 volte, in modo da ottenere una riga lunga 19 caratteri.

50-60	Stampano due caratteri: CHR$(18), che attiva il negativo, quindi uno spazio e CHR$(147), che disattiva il negativo, assieme ad uno spazio.
70-90	Stampano un'altra riga, invertendo l'ordine dei caratteri, in modo da mostrare una vera scacchiera.
110	Cambia il colore nuovamente in azzurro.
120	Fa si che il disegno rimanga sullo schermo senza la scritta READY.

Cambiando il valore della variabile CL, potete cambiare il colore della scacchiera. Come in tutti i programmi, particolarmente in quelli grafici, è importante fare esperimenti. Quanto più operate modifiche, quanto più utilizzate un metodo di programmazione o un comando, tante più scoperte farete.

L'istruzione POKE

Anche se l'istruzione PRINT può essere utilizzata per creare una grande varietà di effetti grafici sul Commodore 64, esiste un altro metodo, molto più versatile e spesso più semplice da usare. Questo metodo è costituito dall'uso della istruzione POKE.

Il chip VIC-II del Commodore 64 aggiorna lo schermo 60 volte al secondo. Non dovete preoccuparvene, viene fatto automaticamente. L'importante è ricordarsi che il chip VIC-II guarda ad una determinata area di memoria per sapere quale dovrebbe essere l'aspetto dello schermo TV. Questo è il modo in cui i vostri caratteri grafici od alfanumerici vengono mostrati quando premete un tasto o mandate in esecuzione un programma. Cambiare un valore numerico in una particolare locazione di memoria, diciamo quella che determina il colore di fondo, dice al chip VIC-II quale colore desiderate. La locazione di memoria che viene esplorata per sapere qual è il colore di fondo è la 53281, mentre il colore della cornice è controllato dalla locazione 53280.

Tavola 2. Valori numerici utilizzati per controllare il colore nelle istruzioni POKE.

Colore	Valore POKE
NERO	0

BIANCO	1
ROSSO	2
BLU-VERDE	3
PORPORA	4
VERDE	5
BLU	6
GIALLO	7
ARANCIO	8
MARRONE	9
ROSSO CHIARO	10
GRIGIO 1	11
GRIGIO 2	12
VERDE CHIARO	13
AZZURRO	14
GRIGIO 3	15

Oltre a leggere le locazioni responsabili del controllo del colore di fondo e della cornice, il chip VIC-II esplora anche altre locazioni di memoria per sapere quale dovrebbe essere l'aspetto dello schermo. Scandisce un'area chiamata *memoria di schermo,* per stabilire quali caratteri mostrare sullo schermo; un'altra serie di locazioni chiamata *memoria colore,* per sapere quali sono i colori da attribuire a ciascun carattere ed un'altra area ancora, *il set di caratteri,* per sapere qual è l'aspetto di ciascun carattere. Controlla anche altre locazioni, ma quelle citate sono le più importanti per quanto riguarda la creazione di grafici con il C64.

Cambiando il contenuto di queste locazioni, usando l'istruzione POKE, potete controllare ciò che appare sullo schermo.

Un comando POKE inserisce un nuovo valore numerico in una locazione di memoria, tramite due numeri separati da virgola. Il primo numero rappresenta la locazione di memoria che si desidera cambiare. Il secondo numero è il *nuovo* valore che si vuole porre nella locazione suddetta. Benché sia possibile inserire, tramite un'istruzione POKE, un qualsiasi numero intero compreso fra 0 e 255 in qualsiasi locazione di memoria da 0 a 65535, esiste un numero limitato di comandi POKE, che userete spesso nella creazione di grafici.

```
POKE 53281,0
```

Questa istruzione POKE, ad esempio, cambierà il colore di fondo

in nero. Per cambiare i colori dello schermo dovete inserire nelle locazioni di memoria opportune un valore numerico compreso fra 0 e 15. Questi valori cambiano i colori all'interno delle istruzioni POKE, proprio come quando avete utilizzato i tasti SHIFT e Commodore per provocare il cambiamento dei colori all'interno di una istruzione PRINT. Per avere un elenco dei valori associati ai colori, e utilizzati nelle istruzioni POKE, vedere la Tavola 2.

Ecco un breve programma, che mostrerà tutte le possibili combinazioni di colore di fondo e di cornice, evidenziando contemporaneamente i valori numerici da inserire nella istruzione POKE per realizzare la combinazione mostrata. Notate che il colore di fondo e quello della cornice, contrariamente ad altri calcolatori, vengono controllati da locazioni di memoria differenti. Il colore di fondo è localizzato in 53281, mentre il colore della cornice è controllato dalla 53280.

Programma 6. Controllo dei colori di fondo e della cornice tramite l'istruzione POKE

```
10 PRINT"[CLR]"
20 FOR BR=0 TO 15
30 FOR BG=0 TO 15
40 POKE 53280,BR
50 POKE 53281,BG
60 PRINT"[HOME][2 GIU'][DESTRA]COLORE
   DELLA CORNICE="BR;"    [SINISTRA] [
   2 DESTRA]COLORE DI FONDO="BG"[SINI
   STRA] "
70 FOR T=0 TO 1000:NEXT
80 NEXT:NEXT
```

Mentre il programma è in esecuzione potrete vedere i valori utilizzati dalle istruzioni POKE. Alcune delle combinazioni di colori non sono particolarmente attraenti, altre non sono molto utili per mostrare dei testi, ma alcune vi piaceranno. Se trovate gradevole una particolare combinazione, limitatevi a premere il tasto RUN/STOP ed a prendere nota dei valori mostrati sullo schermo. Se non riuscite ad identificarli, potete premere i tasti RUN/STOP-RESTORE e quindi battere:

```
PRINT BR <RETURN>
```

e/o

```
PRINT BG <RETURN>
```

e verranno mostrati gli ultimi valori usati (BR è la variabile contenente il valore associato al colore della cornice e BG quella del colore di fondo).

Quando il valore del fondo è 14, o azzurro, sembra che il testo sia scomparso. Le parole e le cifre sono ancora lì, ma sono invisibili, poiché dello stesso colore dello schermo. Questo è uno dei modi in cui i programmatori fanno apparire e scomparire le cose dallo schermo. Se vi capita di mostrare qualcosa sullo schermo con una istruzione PRINT o POKE e questa non appare, la prima cosa da fare consiste nell'inserire un differente valore nella locazione 53281 tramite una POKE; forse, infatti, il carattere è invisibile perché dello stesso colore dello schermo.

Inserimenti sullo schermo tramite le istruzioni POKE

Finora avete creato effetti grafici usando l'istruzione PRINT, che tratta i dati in forma sequenziale. Ogni carattere viene stampato accanto al precedente, a partire da una posizione nota dello schermo. Ogni istruzione PRINT ha un numero opportuno di controlli cursore per piazzare i caratteri sullo schermo, proprio come avete avuto modo di osservare nel disegno del tavolo da bigliardo nei paragrafi precedenti di questo stesso capitolo. Ma questo metodo richiede un lungo tempo per essere programmato e, spesso, parecchi passi.

Un modo più semplice di ottenere questo risultato consiste nell'utilizzare le istruzioni POKE per controllare ogni locazione dello schermo. Questo è il metodo più usato per creare grafici con il Commodore 64.

Le locazioni di memoria rappresentano la chiave per utilizzare le POKE quando create grafici sullo schermo. La memoria del C64 è una lunga serie di indirizzi, uno accanto all'altro. Una parte di questi è usata per la *memoria di schermo.* Dal momento che lo schermo può mostrare 1000 caratteri in una quadrettatura con 40 colonne di larghezza e 25 righe in altezza, esistono 1000 locazioni di memoria riservate al trattamento di ciò che appare sullo schermo.

Ogni locazione di memoria può contenere un numero compreso fra 0 e 255. In altre parole, esistono 256 possibili valori per ogni locazione di memoria. Cambiando il valore numerico, cambiate ciò che compare sullo schermo. In questo modo voi potete scegliere

che cosa far apparire sullo schermo del monitor o del televisore ed anche dove farlo apparire.

Il chip VIC-II legge la memoria di schermo un carattere per volta, partendo dall'angolo superiore sinistro, si muove lungo la riga superiore da sinistra a destra e quindi salta al carattere più a sinistra della riga seguente. Quando raggiunge l'ultimo carattere, nell'angolo inferiore destro, ritorna all'angolo superiore sinistro e ricomincia da capo.

La memoria di schermo sul C64 normalmente inizia alla locazione 1024 e termina alla 2023 (vedi Appendice C). L'angolo superiore sinistro presenta l'indirizzo *più basso,* mentre l'angolo inferiore destro *il più alto.* Il C64 legge da sinistra a destra, dall'alto al basso, proprio come voi. Se ricordate questo fatto, le cose non dovrebbero risultare troppo complicate.

Supponiamo che vogliate piazzare un carattere al centro dello schermo, che è rappresentato dalla colonna 20 della riga 12. Per trovare l'indirizzo esatto di questo punto in memoria moltiplicate il numero di riga (12) per 40, che è il numero totale di locazioni per riga. Il risultato è 480. Quindi aggiungetegli 20, dal momento che volete il ventunesimo carattere (il primo carattere di ogni riga è contraddistinto dallo 0). Il totale è 500, che dovete aggiungere all'indirizzo di partenza, 1024, per ottenere l'indirizzo della locazione di memoria desiderata: 1524. Una semplice formula per effettuare questo calcolo è la seguente:

```
LOCAZIONE DELLA MEMORIA DI SCHERMO=
1024+40*RIGA+COLONNA
```

Utilizzandola potete trovare l'indirizzo di ciascuna delle 1000 locazioni di memoria dello schermo. Per piazzare un carattere nella posizione suddetta tutto ciò che dovete fare è:

```
POKE 1524,81
```

Come per tutte le istruzioni POKE, il primo numero rappresenta la locazione di memoria ed il secondo il nuovo valore che volete inserire in quella locazione. Potete piazzare qualsiasi carattere in una particolare locazione, usando il valore di codice schermo del carattere come secondo numero. Fate riferimento alla tavola dei codici schermo del vostro *Manuale d'uso del Commodore 64* per ottenere i valori suddetti. Nell'esempio precedente verrà mostrato, al centro dello schermo, il carattere grafico «pallino pieno», poiché il suo valore numerico di codice schermo è, appunto, 81. Per far

apparire un altro carattere, come la lettera A, tutto ciò che dovete fare è cambiare il valore sopraccitato in 1 . Se copiate questo esempio e lo mandate in esecuzione, potreste tuttavia non vedere niente sullo schermo. Per ogni locazione di memoria schermo esiste un corrispondente indirizzo di *memoria colore.* Invece di vedere i numeri contenuti in queste locazioni come caratteri, il chip VIC-II interpreta quei numeri come *codici colore.* Questo significa che la memoria colore è l'ombra esatta della memoria di schermo. Potete controllare il colore di un singolo carattere inizializzando la opportuna locazione di memoria colore.

Tuttavia, versioni più recenti del C64, all'atto dell'accensione o quando vengono ri-inizializzati premendo RUN/STOP-RESTORE, riempiono automaticamente la memoria colore con il valore corrispondente al colore di fondo. Quindi, a meno che cambiate il valore corrispondente nella memoria colore quando effettuate una operazione POKE sullo schermo, il carattere che la vostra istruzione POKE inserirà sullo schermo risulterà invisibile (ciò non costituisce un problema quando si usa l'istruzione PRINT, poiché essa si prende automaticamente cura del cambiamento della memoria colore).

Gli indirizzi della memoria colore iniziano in 55296 e continuano per 1000 locazioni di memoria fino a 56295, così come la memoria di schermo occupa 1000 indirizzi. Il chip VIC-II legge la memoria colore nello stesso modo in cui legge la memoria di schermo, dall'angolo superiore sinistro all'angolo inferiore destro. La sola differenza è rappresentata dall'indirizzo della locazione di memoria. Per calcolare l'indirizzo della memoria colore viene usata una formula leggermente differente.

```
LOCAZIONE DI MEMORIA COLORE=
55296+40*RIGA+COLONNA
```

La locazione di memoria colore 55796 rappresenta il centro dello schermo ed è accoppiata alla locazione di memoria schermo 1524. Per cambiarne il colore tutto ciò che dovete fare è inserire un valore da 0 a 15 (gli stessi valori che avete usato per cambiare i colori della cornice e del fondo), tramite una istruzione POKE, in quella locazione. Potete farlo in questo modo:

```
10 POKE 1524,81
20 POKE 55796,0
```

Ciò farà apparire il carattere «pallino pieno» in nero al centro dello schermo.

Usando questo metodo, consistente nell'inserire tramite POKE caratteri e colori direttamente sullo schermo, potete creare suppergiù qualsiasi disegno desideriate. Benché possa sembrare un lungo lavoro di battitura, è più breve che usare il controllo cursore e l'istruzione PRINT. La maggior parte dei programmatori usa il metodo delle POKE, quando crea grafici sul C64.

Una dimostrazione dell'uso delle POKE può variare da qualcosa di semplice a qualcosa di piuttosto elaborato. Ad esempio, creare una cornice attorno allo schermo è piuttosto semplice. Il programma seguente lo fa.

Programma 7. Cornice

```
10 SC=1024:CL=55296:PRINT"[CLR]"
20 POKE 53281,1
30 RIGA=0:FOR COLONNA=0 TO 39:GOSUB 80:
   NEXT
40 COLONNA=0:FOR RIGA=0 TO 24:GOSUB 80:
   NEXT
50 RIGA=24:FOR COLONNA=0 TO 39:GOSUB 80
   :NEXT
60 COLONNA=39:FOR RIGA=0 TO 24:GOSUB 80
   :NEXT
70 GOTO 70
80 POKE CL+COLONNA+RIGA*40,0:POKE SC+CO
   LONNA+RIGA*40,102:RETURN
```

Commenti al programma

Riga	**Commento**
10	Inizializza i valori di SC e CL, rispettivamente la locazione di partenza per la memoria di schermo e la memoria colore.
20	Cambia il colore di fondo in bianco.
30-60	Inizializzano i bordi. Per primo il bordo superiore in riga 30, quindi il sinistro con la riga 40, seguito dal bordo inferiore e dal destro alle righe 50 e 60.
70	Trattiene in ciclo il programma, in modo che la scritta READY non rovini l'uscita.
80	Inserisce il colore ed il carattere, tramite una POKE, per

ogni locazione dello schermo.

Questo è solo un breve programma di grafica, ma il suo effetto vi stupirà. Potete cambiare il colore della cornice, ed il carattere usato per tracciarla, cambiando semplicemente i valori che vengono utilizzati dalla istruzione POKE in riga 80. Fate esperimenti per vedere la differenza.

Effetti grafici iniziali

Ora avete un'idea, benché relativamente limitata, delle capacità grafiche del Commodore 64. Le possibilità di scelta nella creazione di grafici sono numerose. Potete usare istruzioni PRINT o POKE per creare questi effetti grafici. Potete persino usare i valori di codice simbolico CHR$ per far apparire caratteri alfanumerici e grafici sullo schermo.

Ma non state ancora realizzando quelle schermate tipo videogames, che tanto vi hanno colpito. Altri articoli in questo libro vi mostrano come farlo. Ad esempio, potete vedere come creare caratteri grafici personalizzati al Capitolo 3, ad esempio, o come disegnare ed utilizzare gli sprite del C64 ai Capitoli 1 e 2 del volume 2.

Ricordate che state imparando un nuovo linguaggio, il BASIC, che, come qualsiasi altro linguaggio, richiede pratica e tempo per essere utilizzato in maniera disinvolta. *Ce la farete* col tempo. La vostra reazione iniziale di ansietà ed abbattimento scomparirà facendo esperimenti con il calcolatore, trovando nuove idee per i vostri programmi.

Il Commodore 64 *è* un computer grafico. Ha solo bisogno di voi.

Capitolo uno

I caratteri grafici

C. Regena

Un modo di tracciare grafici sullo schermo consiste nell'utilizzare il set di caratteri grafici incorporato.

Si possono realizzare grafici (e disegni) con i simboli che trovate sulla tastiera del C64. Noterete che ciascun tasto ha un simbolo sulla faccia superiore. Questo simbolo è quello che si ottiene premendo il tasto stesso. Ora guardate la faccia anteriore del tasto. Parecchi tasti hanno due simboli racchiusi in quadrati. Questi sono i simboli utilizzati per i grafici.

Premete SHIFT ed un tasto contemporaneamente ed otterrete il simbolo sulla destra della faccia anteriore del tasto. All'estrema sinistra della tastiera, nella fila di tasti inferiore, c'è un tasto con il simbolo Commodore, chiamato tasto Commodore. Provate a premerlo unitamente ad un tasto con i due simboli grafici sulla faccia anteriore. Sullo schermo apparirà il simbolo che appare alla sinistra. Ad esempio, prendete in considerazione il tasto «S». Se lo premete da solo sullo schermo apparirà una S. Se premete il tasto «S» e SHIFT, apparirà un cuoricino. Se lo premete congiuntamente al tasto Commodore, apparirà il simbolo grafico ⌐.

Come spostare il cursore

Per tracciare un disegno sullo schermo non è nemmeno necessario che sappiate programmare. Innanzitutto premete SHIFT e CLR/HOME per cancellare lo schermo. Il cursore (il quadratino lampeggiante che vi mostra dove state scrivendo) si troverà nell'angolo superiore sinistro. Ora potete iniziare a tracciare un grafico o a disegnare, usando i tasti SHIFT e Commodore accoppiati agli altri tasti per ottenere il simbolo grafico voluto.

Il cursore, naturalmente, si muove da sinistra a destra attraverso lo schermo. Quando raggiunge il termine di una riga, si sposta all'inizio della riga successiva. Per spostare il cursore in una traiettoria differente usate i tasti di controllo cursore. Questi due tasti si trovano all'estrema destra della sezione principale della tastiera, sulla riga inferiore. Sono contraddistinti dalla scritta CRSR e da

frecce.

È possibile spostare il cursore verso destra sia premendo la barra spaziatrice che il tasto CRSR di destra. Quest'ultimo ha frecce verso destra e verso sinistra. La differenza tra questi due metodi consiste nel fatto che la barra spaziatrice inserisce spazi mentre si muove e quindi cancellerà tutto ciò che trova sul suo cammino, mentre il tasto CRSR si sposta senza apportare cambiamenti a ciò che compare sullo schermo. Per spostarvi a sinistra premete SHIFT e lo stesso tasto CRSR.

Ora provate a premere il tasto con le frecce rivolte verso l'alto e verso il basso. Se premete solo il tasto CRSR, il cursore scenderà. Se volete salire, premete SHIFT ed il tasto CRSR. Ancora una volta non cancellerete niente di ciò cui passerete sopra.

Come si traccia un disegno

Per inserire un disegno in un programma BASIC potete usare le istruzioni PRINT e copiare ciò che avete disegnato sullo schermo. Per aiutarvi potete schizzare il vostro disegno su di un foglio di carta da lucido. Lo schermo ha una larghezza di 40 colonne per 25 righe.

Per iniziare cancellando lo schermo, usate una istruzione del tipo

```
10 PRINT "[CLR]"
```

Battete il numero di riga, il comando PRINT, i doppi apici, quindi premete contemporaneamente i tasti SHIFT e CLR/HOME (apparirà un simbolo che significa CLEAR e che ha l'aspetto di un cuoricino in negativo) e nuovamente i doppi apici, per chiudere.

Continuate usando istruzioni PRINT con i simboli desiderati inclusi nei doppi apici. Potete anche usare i tasti di controllo cursore internamente ai doppi apici per segnalare al calcolatore di spostare il cursore mentre sta disegnando in una posizione diversa. Ecco un esempio (fate riferimento all'Appendice B per le convenzioni del listato):

```
10 PRINT "[CLR]"
20 PRINT "A"
30 PRINT "[3 A DESTRA]S"
40 PRINT "[3 A DESTRA][3 GIU']X"
50 PRINT "Z"
60 PRINT
```

```
70 PRINT "O[(3 Y)]P[GIU'][A SINISTRA]N
   [GIU'][2 A SINISTRA]N[GIU'][2 A SIN
   ISTRA]N"
```

La riga 40 indica di premere il tasto CRSR di destra tre volte, quindi il tasto CRSR di sinistra tre volte, poi SHIFT e la lettera X. La riga 70 indica di premere SHIFT e la lettera O, quindi il tasto Commodore e la lettera Y tre volte, il tasto CRSR di sinistra, il tasto CRSR che controlla gli spostamenti del cursore verso sinistra (vale a dire SHIFT ed il tasto CRSR di destra), SHIFT e la lettera N, il tasto CRSR di sinistra, e cosi via.

Se usate il tasto CLR/HOME senza premere SHIFT, il cursore tornerà inizialmente nella posizione di «home», a sinistra della riga superiore dello schermo, ma il contenuto dello schermo non verrà cancellato. Se volete che sullo schermo, assieme al vostro disegno, non appaia la scritta READY, usate una riga del tipo 80 GOTO 80 per mantenere il programma in ciclo. Per interromperlo premete il tasto RUN/STOP.

Come si aggiunge il colore

Vediamo ora come aggiungere il colore ai vostri disegni. Premete il tasto CTRL ed uno dei numeri sulla riga superiore della tastiera, quindi cominciate a premere i tasti. Ora avete a disposizione un nuovo colore. Il Commodore 64 ha altri otto colori. Per ottenere ciascun colore premete il tasto Commodore accoppiato ad uno dei tasti colore numerati. Potete utilizzare questi tasti colore all'interno delle istruzioni PRINT nei vostri programmi. Dal momento in cui usate un tasto colore tutto ciò che viene stampato avrà quel colore finché verrà nuovamente cambiato.

Due altri tasti particolarmente utili nella grafica su schermo sono i tasti RVS. Per ottenere RVS ON, che ha la funzione di mostrare in negativo ogni lettera o carattere grafico seguente, premete CTRL e 9 simultaneamente. Premete ad esempio SHIFT e Q. Vedrete apparire un cerchio pieno. Ora provate a premere CTRL e 9, per ottenere RVS ON, quindi SHIFT e Q. Il cerchio, ora, apparirà nel colore di fondo (blu), mentre attorno ad esso avrà una cornice nel colore cursore (azzurro). Per rientrare nel formato normale premete CTRL e 0 (zero) per ottenere RVS OFF. Le convenzioni per i listati all'interno delle istruzioni PRINT sono [RVS] per RVS ON e [OFF] per RVS OFF.

Per ottenere delle barre colorate potete usare il comando RVS ON e la barra spaziatrice. Prendete in considerazione il tasto '*'. Il simbolo alla sinistra della faccia anteriore si ottiene premendo il tasto Commodore e l'asterisco. Supponiamo che desideriate il triangolo inferiore invece del superiore, tuttavia con la suddivisione del carattere presentata da questo tasto, non in quella del carattere grafico corrispondente al tasto '\'. Premete RVS ON quindi Commodore e '*'.

Il programma 1 mostra come si può tracciare un istogramma a barre partendo da dei dati, in modo da rendere le statistiche più interessanti. Questo programma mostra l'uso dei tasti colore e dei tasti RVS ON e RVS OFF.

La funzione TAB viene usata con le istruzioni PRINT per iniziare a stampare a partire da una certa colonna. Questa funzione equivale a premere più volte il tasto CRSR di destra. PRINT TAB(10);«X» ha lo scopo di stampare la lettera X in colonna 10.

Grafici con le istruzioni POKE

Oltre a creare grafici tramite le istruzioni PRINT, è anche possibile farlo usando la istruzione POKE. Usate la «Tavola delle locazioni di memoria dello schermo» (Appendice C) per inserire, tramite una istruzione POKE, in una certa locazione di schermo un carattere contraddistinto da un numero ricavabile dalla «Tavola dei codici di schermo» (Appendice G).

Noterete che la Tavola delle locazioni di memoria dello schermo contiene numeri da 1024 a 2023. Supponiamo di voler inserire un asterisco ('*') nella decima colonna della terza riga. Dalle tavole possiamo rilevare che la riga inizia con il numero 1104. Aggiungiamoci 10 per posizionarci sulla colonna desiderata ed otteniamo la locazione di memoria 1114. Ora guardiamo la tavola dei codici di schermo. L'asterisco nella colonna della serie 1 corrisponde al numero 42 nella colonna POKE. Il comando BASIC opportuno è quindi POKE 1114,42. Per aggiungere il colore al carattere suddetto si può sia far riferimento alla «Tavola delle locazioni di memoria colore» (Appendice D) sia limitarsi ad aggiungere 54272 al numero della Tavola delle locazioni di memoria dello schermo. Scegliete un numero di colore compreso tra 0 e 15. Il comando necessario per ottenere un asterisco rosso sarebbe POKE55386,2 (per una spiegazione più completa vedi l'articolo successivo «Grafica tramite l'istruzione POKE»).

Capitolo uno

Per vedere quanto rapidamente si può spostare un cerchio attraverso lo schermo, usando il metodo POKE, provate questo programma:

```
10 FOR L=1824 TO 1903
20 POKE L,87:POKE L+54272,7
30 POKE L,32
40 NEXT L
```

Il ciclo FOR-NEXT fa variare da 1824 a 1903 la variabile L, che rappresenta la locazione di memoria schermo. La riga 20 inserisce un cerchio nella locazione suddetta, tramite una istruzione POKE, quindi assegna al cerchio il colore giallo. La riga 30 cancella il cerchio inserendo uno spazio (carattere 32) nella locazione. Al variare dell'indice del ciclo la locazione cambia di una unità video.

Un vantaggio della grafica tramite le istruzioni POKE consiste nella possibilità di specificare esattamente la locazione di schermo. Quando usate le istruzioni PRINT per realizzare la grafica dovete sapere dove la precedente istruzione PRINT ha lasciato il cursore o dove sarà la successiva. Quando disegnate grafici in un certo ordine, potrebbe essere preferibile usare istruzioni PRINT per realizzare parte del disegno e istruzioni POKE per inserire grafici all'interno del disegno realizzato nel modo suddetto.

Un esempio pratico

Il programma 2, che insegna le posizioni base per la digitazione, mostra come un programma didattico possa essere arricchito dai grafici. Le mani vengono disegnate usando istruzioni PRINT ed i simboli grafici riportati sui tasti. Le lettere che compaiono al di sopra delle dita vengono inserite nelle locazioni opportune tramite istruzioni POKE. Parecchie delle istruzioni PRINT dimostrano l'utilità dell'uso dei tasti di controllo cursore per posizionare le scritte. La funzione TAB viene usata in parecchi punti al posto dei tasti cursore quando si rendono necessari rilevanti spostamenti.

Commenti al programma

Riga	**Commento**
2	Cambia il colore dello schermo in bianco
3	Dei comandi POKE inizializzano i registri degli effetti

sonori. Vengono definite le variabili F1, F2 e W per successivi usi all'interno di comandi per effetti sonori.

4 Definisce variabili alfanumeriche per stampare i grafici.

6-8 Leggono da istruzioni DATA il contenuto delle seguenti variabili a più dimensioni:
P(I), locazioni di schermo necessarie per inserire le lettere, tramite POKE, sopra le dita appropriate;
P$(I), lettera dell'alfabeto;
L(I), numeri di codice necessari per inserire lettere o simboli sullo schermo tramite POKE;
S(I) e T(I), numeri coinvolti negli effetti sonori.

9 Richiama il programma principale, saltando i sottoprogrammi seguenti.

10-150 Sottoprogramma per cancellare lo schermo e disegnare le mani.

200-220 Sottoprogramma di ricerca del tasto premuto e di controllo della correttezza del tasto stesso. Se è stato premuto il tasto giusto, viene rimpiazzato sullo schermo da uno spazio (cioè cancellato); altrimenti il calcolatore attende che venga premuto il tasto giusto.

400-480 Stampano l'intestazione ed attendono che l'utente prema un tasto.

500 Richiama il sottoprogramma che disegna le mani.

510-560 Fanno risuonare un campanello e stampano una lettera sopra ogni dito.

570-610 Mostrano le istruzioni per il primo esercizio.

620-640 Cancellano le lettere che appaiono sopra le dita.

650-710 Propongono un esercizio di digitazione. Le lettere vengono presentate per tre volte, da sinistra a destra, in ordine. Risuona il campanello e la lettera od il simbolo vengono stampati sopra il dito corrispondente. La riga 690 richiama il sottoprogramma che controlla se un tasto viene premuto. Perché il programma prosegua deve essere premuto il tasto esatto.

720-780 Scelgono a caso le lettere da proporre come esercizio.

790-820 Offrono l'opportunità di ripetere l'esercizio o proseguire nel programma e richiamano le parti appropriate

	del programma.
830	Cancella lo schermo.
840	Ripristina i dati, nel caso in cui l'esercizio venga ripetuto.
850	Legge i primi 40 termini che sono stati usati in precedenza e che non vengono usati in questo esercizio.
860-880	Leggono dalle istruzioni DATA nove parole e frasi, inserendole nel vettore A$ perché siano utilizzate nell'esercizio.
890-1220	Ripropongono l'esercizio finché non vengono completate correttamente cinque frasi.
900-910	Mostrano le istruzioni.
920	Sceglie una frase a caso. Se la frase è stata battuta correttamente, viene posta a "" (stringa vuota) e deve essere scelta un'altra frase.
930	Inizializza la variabile alfanumerica B$ e mostrà la frase da copiare.
940	Posiziona la stampa per la digitazione dell'utente.
950-1000	Mostrano il tasto premuto o escono dal ciclo (se viene premuto RETURN). B$ contiene ciò che l'utente ha battuto.
1010	Confronta la frase battuta con quella assegnata dall'esercizio.
1020-1100	Se la frase non è corretta, fanno risuonare un cicalino e stampano ERRATO, quindi attendono che l'utente prema RETURN per proporre un'altra frase.
1110	Se la risposta è corretta, stampa un cuore rosso. Il numero di cuori rossi rappresenta il numero di frasi corrette.
1120-1200	Suonano un motivo per le risposte corrette.
1210-1220	Pongono la frase A$ a "" (stringa vuota), in modo che non venga scelta nuovamente; riciclano per una nuova frase.
1230-1280	Offrono l'opportunità di ripetere l'esercizio di digitazione delle lettere o quello delle frasi o di concludere il programma e richiama le opportune sezioni di programma.
1290-1300	Cancellano lo schermo e concludono il programma.

Programma 1. Istogramma a barre

```
10 PRINT"{CLR}{WHT}"
20 PRINTTAB(14);"POPOLAZIONE"
30 PRINTTAB(16);"{GIU'}{RVS}{YEL}
   { 2 SPAZI}{OFF}{WHT} 1970"
40 PRINTTAB(16);"{RVS}{RED}{ 2 SPAZI}{OFF}
   {WHT} 1980{GIU'}"
50 FORC=1TO5
60 READS$,P1,P2
70 PRINT"{GIU'}";S$;TAB(10);
80 FORI=1TOINT(P1/75000+.5)
90 PRINT"{RVS}{YEL} ";
100 NEXTI
110 PRINT"{OFF}{WHT}";TAB(38-LEN(STR$(P1))
    );P1
120 PRINTTAB(10);
130 FORI=1TOINT(P2/75000+.5)
140 PRINT"{RVS}{RED} ";
150 NEXTI
160 PRINT"{OFF}{WHT}";
170 PRINTTAB(38-LEN(STR$(P2)));P2
180 NEXTC
190 DATANEVADA,488738,799184,UTAH,1059273,
    1461037,WYOMING,332416,470816
200 DATAIDAHO,713015,943935,MONTANA,694409
    ,786690
210 GOTO210
220 END
```

Programma 2. Esercizi di digitazione

```
2 POKE53281,1
3 POKE54296,15:POKE54277,8:POKE54278,8:F1=
  54273:F2=54272:W=54276
4 F$="U*I":G$="B -":H$=G$+G$+G$
6 FORI=1TO8:READP(I),P$(I),L(I),S(I),T(I):
  NEXT
```

```
7 DATA1467,A,1,34,75,1390,S,19,38,126,1353
  ,D,4,43,52,1396,F,6,45,198
8 DATA1411,J,10,51,97,1374,K,11,57,172,141
  7,L,12,64,188,1500,":",58,68,149
9 GOTO400
10 PRINT"{CLR}{ 10 GIU'}{RED}"
20 PRINTTAB(8);F$;TAB(29);F$
30 PRINTTAB(5);F$;G$;F$;TAB(26);F$;G$;F$
40 PRINTTAB(5);H$;TAB(26);H$
50 PRINTTAB(5);H$;TAB(26);H$
60 PRINT"{ 2 SPAZI}";F$;H$;TAB(26);H$;F$
70 FORI=1TO3
80 PRINT"{ 2 SPAZI}";G$;H$;TAB(26);H$;G$
90 NEXTI
100 PRINT"{ 2 SPAZI}B JK JK JK - {RVS}
    {BLK}{ 2 SPAZI}SPAZIO{ 2 SPAZI}{OFF}
    {RED} B JK JK JK -"
110 PRINT"{ 2 SPAZI}B";TAB(13);"-";TAB(26)
    ;"B";TAB(37);"-"
120 PRINT"{ 2 SPAZI}B";TAB(13);"- N[<Y>]P
    { 4 SPAZI}O[<Y>]M B";TAB(37);"-"
130 PRINT"{ 2 SPAZI}B";TAB(13);"-N
    { 2 SPAZI}[<M>]{ 4 SPAZI}[<G>]
    { 2 SPAZI}MB";TAB(37);"-"
140 PRINT"{ 2 SPAZI}B";TAB(17);"N
    { 4 SPAZI}M";TAB(37);"-"
150 RETURN
200 GETE$:IFE$<>P$(J)THEN200
210 POKEP(J),32
220 RETURN
400 PRINT"{CLR}"
410 PRINTTAB(16);"{ 3 GIU'}ESERCIZI"
420 PRINTTAB(13);"{ 2 GIU'}DI DITEGGIATURA
    "
430 PRINTTAB(12);"{ 2 GIU'}POSIZIONI DI BA
    SE"
440 PRINT"{ 6 GIU'} ORA APPARIRA' IL DIAGR
    AMMA DELLE MANI."
450 PRINT" METTI LE DITA SUI TASTI COME IN
```

```
    DICATO."
460 PRINT"{ 2 GIU'}{ 6 SPAZI}PREMI <RETURN
    > PER INIZIARE."
470 GETE$:IFE$=""THEN470
480 IFASC(E$)<>13THEN470
500 GOSUB10
510 FORI=1TO8
520 POKEF1,S(I):POKEF2,T(I):POKEW,17
530 POKEP(I),L(I):POKEP(I)+54272,6
540 FORD=1TO100:NEXT
550 POKEF1,0:POKEF2,0:POKEW,0
560 NEXTI
570 PRINT"{HOME}{ 6 SPAZI}METTI LE DITA IN
     POSIZIONE."
580 PRINT"{GIU'} PREMI QUALSIASI TASTO PER
     CONTINUARE."
590 GETE$:IFE$=""THEN590
600 PRINT"{HOME}{ 6 SPAZI}PREMI L'ULTIMO T
    ASTO APPARSO"
610 PRINT"{GIU'}{ 38 SPAZI}"
620 FORI=1TO8
630 POKEP(I),32
640 NEXTI
650 FORI=1TO3
660 FORJ=1TO8
670 POKEFI,S(J):POKEF2,T(J):POKEW,17
680 POKEP(J),L(J)
690 GOSUB200
700 POKEW,0
710 NEXTJ,I
720 FORI=1TO30
730 J=INT(RND(0)*8)+1:IFJ=KTHEN730
740 K=J:POKEF1,S(J):POKEF2,T(J):POKEW,17
750 POKEP(J),L(J)
760 GOSUB200
770 POKEW,0
780 NEXTI
790 PRINT"{HOME}SCEGLI:{ 2 SPAZI}1 RIPROVA
    RE{ 18 SPAZI}"
```

```
800 PRINTTAB(9);"2 PROSEGUIRE COL PROGRAMM
    A"
810 GETE$:IFE$="1"THEN500
820 IFE$<>"2"THEN810
830 PRINT"{CLR}"
840 RESTORE
850 FORI=1TO40:READE$:NEXT
860 DATA"ASA :L:","DFD KJK","ADA :K:","AFA
     :J:"
870 DATA"SDS LKL","SFS LJL","DFD KJK","DAL
    LA SALA","LA FALLA"
880 FORI=1TO9:READA$(I):NEXT
890 FORI=1TO5
900 PRINT"{CLR}{ 7 SPAZI}BATTI L'ESERCIZIO
     MOSTRATO"
910 PRINT"{ 12 SPAZI}E PREMI <RETURN>.
    { 8 GIU'}"
920 J=INT(9*RND(0))+1:IFA$(J)=""THEN920
930 B$="":PRINTTAB(14);A$(J)
940 PRINTTAB(14);
950 FORK=1TO20
960 GETE$:IFE$=""THEN960
970 IFASC(E$)=13THEN1010
980 PRINTE$;
990 B$=B$+E$
1000 NEXTK
1010 IFB$=A$(J)THEN1110
1020 POKEF1,43:POKEF2,52:POKEW,17
1030 FORD=1TO100:NEXT
1040 POKEF1,34:POKEF2,75:POKEW,17
1050 FORD=1TO100:NEXT:POKEW,0
1060 PRINT:PRINTTAB(15);"{ 3 GIU'}SBAGLIAT
     O"
1070 PRINTTAB(12);"{GIU'}PREMI{ 2 SPAZI}<R
     ETURN>"
1080 GETE$:IFE$=""THEN1080
1090 IFASC(E$)<>13THEN1080
1100 GOTO900
```

```
1110 FORD=1TOI:POKE1600+D,83:POKE1600+D+54
     272,2:NEXT
1120 POKEF1,34:POKEF2,75:POKEW,17
1130 FORD=1TO100:NEXT:POKEW,0
1140 POKEF1,43:POKEF2,52:POKEW,17
1150 FORD=1TO100:NEXT:POKEW,0
1160 POKEF1,51:POKEF2,97:POKEW,17
1170 FORD=1TO100:NEXT:POKEW,0
1180 POKEF1,68:POKEF2,149:POKEW,17
1190 FORD=1TO300:NEXT
1200 POKEW,0
1210 A$(J)=""
1220 NEXTI
1230 PRINT:PRINT"{ 5 GIU'}SCEGLI:
     { 2 SPAZI}1 ESERCIZI CON LE LETTERE"
1240 PRINTTAB(9);"2 ESERCIZI CON LE PAROLE
     "
1250 PRINTTAB(9);"3 FINE"
1260 GETE$:IFE$="1"THEN500
1270 IFE$="2"THEN830
1280 IFE$<>"3"THEN1260
1290 PRINT"{CLR}"
1300 END
```

Grafica per mezzo dell'istruzione POKE

C. Regena

Si può realizzare grafica sullo schermo sia tramite le istruzioni PRINT che le istruzioni POKE. Il metodo POKE risulta particolarmente utile per le animazioni.

Il formato del comando POKE è POKEn1,n2, dove n1 è un indirizzo di memoria e n2 è un valore numerico. Provate il comando POKE 53280,n2 per cambiare il colore della cornice e POKE 53281,n2 per cambiare il colore dello schermo, dove n2 è un numero qualsiasi compreso tra 0 e 15.

Proviamo qualcuno di questi valori:

```
POKE 53281,12
POKE 53280,1
```

Per ritornare al formato normale limitatevi a premere RUN/STOP e RESTORE, oppure a battere POKE 53280,14 e POKE 53281,6.

Ecco un programma che consente di vedere tutte le combinazioni possibili:

```
10 FOR I=0 TO 15
15 POKE 53281,I: REM DECIDE IL COLORE
   DELLO SCHERMO
20 FOR J=0 TO 15
30 POKE 53280,J: REM DECIDE IL COLORE
   DELLA CORNICE
40 FOR D=1 TO 200: NEXT D
50 NEXT J,I
```

Capitolo uno

Grafici semplici

Proviamo a mettere qualche carattere grafico sullo schermo. Fate riferimento alla Appendice C.

La quadrettatura rappresenta lo schermo del C64, di 25 righe di 40 colonne ciascuna. Ognuno dei numeri che contraddistinguono una particolare locazione di memoria si ottiene sommando i numeri di riga e di colonna. Questo è il numero n1 di cui avete bisogno come locazione di memoria della istruzione POKE. Ad esempio, per inserire un carattere con una istruzione POKE in riga 10, colonna 4, useremo una variabile numerica n1 di 1384+4=1388.

Fate riferimento alla Appendice G per una Tavola dei codici caratteri per il numero n2 da utilizzarsi nei comandi POKE. Cercate sotto la colonna titolata Serie 1 il simbolo che volete stampare. Troverete il numero corrispondente sotto la colonna a titolo POKE. Ad esempio, per ottenere il seme di picche, il numero è 65.

Ora avete tutti i parametri necessari per una istruzione POKE. Poniamo un seme di picche in riga 10, colonna 4. Ora sappiamo che il comando corrispondente è POKE 1388,65.

L'unico problema consiste nel fatto che, se tracciaste grafici in questo modo, non sareste in grado di vederli (tranne che sulle prime versioni del C64). Ciò dipende dal fatto che i caratteri grafici che avete inserito sullo schermo, tramite le istruzioni POKE, hanno lo stesso colore dello sfondo dello schermo, il che rende il carattere invisibile. Una soluzione consiste nel cambiare il colore dello schermo dopo aver inserito i caratteri grafici tramite una istruzione POKE.

Ad esempio:

```
10 PRINT"[CLR]"
20 POKE 1388,65: REM DISEGNA UN SEME
   DI PICCHE IN BIANCO
30 POKE 53281,2: REM CAMBIA IL COLORE
   DELLO SCHERMO IN ROSSO
40 GOTO 40
```

Premete il tasto RUN/STOP per interrompere il programma. Premete contemporaneamente RUN/STOP e RESTORE per tornare alle normali condizioni di schermo.

Capitolo uno

Come si cambiano i colori

Supponiamo che vogliate mantenere il normale colore di schermo pur disegnandovi dei grafici. Potete cambiare il colore dei caratteri tracciati inserendo un codice colore in una opportuna locazione di memoria colore tramite una istruzione POKE. Fate riferimento all'Appendice D. Vi troverete una Tavola delle locazioni di memoria dei codici colore. Ogni locazione di schermo è contraddistinta da un numero (ottenuto aggiungendo i numeri di riga e di colonna mostrati) ed ospita un registro colore; questo valore numerico corrisponde alla variabile n1 della nostra istruzione POKE. I codici colore sono elencati nella Appendice E. Questo codice colore rappresenterà il valore numerico n2 per la nostra istruzione POKE.

Ad esempio, poniamo lo stesso seme di picche in riga 10, colonna 4. Troviamo il valore numerico di memoria colore corrispondente alla locazione di schermo 1388. Contando dieci righe a partire dal margine superiore, troverete il valore numerico 55656. Aggiungendo il valore di colonna 4, otteniamo 55660. Notate che la differenza tra le locazioni corrispondenti di schermo e di memoria colore sarà sempre uguale a 54272.

Quindi, per inserire un seme di picche rosso sullo schermo possiamo utilizzare il seguente programma:

```
10 PRINT"[CLR]"
20 POKE 1388,65
30 POKE 55660,2
```

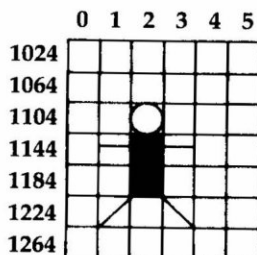
Potete far lampeggiare un carattere sullo schermo, cambiandone il codice colore. Provate il programma seguente:

```
10 PRINT"[CLR]"
20 POKE 1388,65
25 FOR C=1 TO 20
26 POKE 55660,6
27 FOR D=1 TO 100:NEXT D
28 POKE 55660,1
29 FOR D=1 TO 100:NEXT D
35 NEXT C
```

Capitolo uno

Ora siete pronti per schizzare un vostro disegno e disegnarlo tramite valori di POKE. Ecco un programma esemplificativo:

```
5 POKE 53281,1: REM SCHERMO BIANCO
10 PRINT"[CLR]"
12 L=54272
14 POKE 1106,87:POKE 1106+L,2
16 POKE 1146,102:POKE 1146+L,6
18 POKE 1186,102:POKE 1186+L,6
20 POKE 1145,64:POKE 1145+L,6
22 POKE 1147,64:POKE 1147+L,6
24 POKE 1225,78:POKE 1225+L,6
26 POKE 1227,77:POKE 1227+L,6
28 GOTO 28
```

0 1 2 3 4 5
1024
1064
1104
1144
1184
1224
1264

Per provare ad animare il disegno cambiatelo, inserendo caratteri differenti tramite istruzioni POKE o tracciando e cancellando caratteri per spostarlo. Cambiate il programma precedente aggiungendo le righe seguenti: riuscirà il nostro pupazzetto a volare?

```
28 FOR I=1 TO 50
30 POKE 1145,99
32 POKE 1147,99
34 POKE 1145,64
36 POKE 1147,64
38 NEXT I
40 GOTO 40
```

Il set di caratteri

Per realizzare disegni sono disponibili due serie di caratteri, ma può apparire sullo schermo una sola serie per volta. Probabilmente avrete già scoperto che, se avete già dei caratteri sullo schermo e premete contemporaneamente i tasti Commodore e SHIFT, tutte le lettere maiuscole si trasformano in minuscole. La prima condizione rappresenta il set di caratteri numero 1 e la seconda il numero 2.

Prima di cominciare a tracciare i vostri disegni un comando POKE 53272,23 vi metterà a disposizione la seconda serie di caratteri e POKE 53272,21 vi rimetterà a disposizione la prima.

Anche i caratteri negativi sono disponibili. Il valore numerico che

contraddistingue il negativo di ciascun carattere è calcolabile aggiungendo 128 al valore numerico ricavabile dalle tavole per il simbolo in positivo.

Potete utilizzare il comando PEEK per sapere quale carattere si trova in una particolare locazione di memoria o quale ne sia il colore. Potete utilizzare il comando PEEK, ad esempio, per rilevare un ostacolo od una collisione nei giochi. PEEK(n) vi fornirà il valore numerico contenuto nella locazione di memoria n. Alcune forme valide per il comando in questione sono le seguenti:

```
PRINT PEEK(7911)
200 IF PEEK(A)=32 THEN 350
```

In un primo momento può sembrare che l'istruzione PEEK non funzioni con le locazioni di memoria colore, dal momento che si ottengono dei valori differenti da quelli che avete inserito tramite delle istruzioni POKE. Per ovviare all'inconveniente usate:

```
X=PEEK(n)AND15
```

invece di:

```
X=PEEK(n)
```

Dovete solo ricordarvi di questa semplice modifica quando n rappresenta una locazione di memoria colore.

Per fornire una ulteriore dimostrazione delle potenzialità grafiche della istruzione POKE possiamo analizzare i due programmi esemplificativi sottoindicati. Nel programma 1, I e J rappresentano una coppia di coordinate che identificano la locazione di memoria occupata dalla palla. La palla rimbalza quando colpisce la cornice.

Effetti grafici nei giochi

Il programma 2 mostra come sia possibile creare disegni tramite la funzione POKE ed animarli per un gioco. «Difenditi!» è un gioco spaziale per un solo giocatore. Siete posizionati sul margine sinistro dello schermo e dovete difendere il vostro territorio: non dovete lasciare che gli invasori spaziali provenienti dal margine destro dello schermo raggiungano la vostra frontiera!

Allineatevi alla stessa quota di uno degli invasori, premendo ^ per

alzarvi ed il tasto CRSR di sinistra per abbassarvi, quindi fate fuoco premendo la barra spaziatrice od il tasto funzione f7. Guadagnate dieci punti per ogni invasore abbattuto, ma ne perdete cinque ogni volta che ne mancate uno.
Dopo aver giocato un paio di volte con questo programma provate a modificarlo, per adattarlo ai vostri gusti. Usate caratteri grafici e colori differenti. Cambiate il movimento da orizzontale in verticale. Cambiate il meccanismo del punteggio. Ad esempio, dopo aver raggiunto un determinato punteggio si potrebbe far variare l'aspetto e la velocità degli invasori.

Commenti al programma 2

Riga	**Commento**
1	Inizializza le variabili TS per registrare il punteggio più alto ed O per lo scarto tra il valore numerico degli indirizzi di memoria schermo e quelli di memoria colore.
2	Definisce la funzione R(X), la quale calcola il valore numerico che contraddistingue la locazione di memoria associata ad una riga casuale; salta a riga 200.
10	Cancella lo schermo; inizializza i colori di schermo e della cornice. Inizializza altre variabili. N è la posizione della vostra astronave, SC è il punteggio e D il livello di difficoltà.
20	Piazza l'astronave dei difensori sullo schermo.
22-25	Piazzano casualmente gli invasori, assicurandosi che non si trovino sulla stessa riga dei difensori.
30	Rileva quale tasto sia stato premuto. Se è uno dei tasti che controllano il fuoco, salta a riga 60.
32-34	Se viene premuto uno dei tasti che controllano il movimento, spostano l'astronave nella direzione voluta.
35	Incrementa la variabile L per determinare la velocità degli invasori.
36	Aggiorna la posizione degli invasori, spostandoli verso sinistra di una unità video.
37-42	Se un invasore raggiunge il margine sinistro dello schermo, saltano alla riga 100 per concludere il gioco.
44-50	Spostano gli invasori; riciclano per acquisire il prossimo tasto premuto.

62-68	Controllano la posizione degli invasori per verificare se uno di essi è stato colpito.
70	Decrementa il punteggio di cinque, se il colpo è andato a vuoto.
72-78	Procedura richiamata nel caso in cui l'invasore sia stato colpito; scelgono la nuova posizione degli invasori.
80	Aumenta di dieci il punteggio; cancella l'invasore colpito.
82-84	Stampano il punteggio e riciclano per acquisire il prossimo tasto premuto.
90-94	Controllano la posizone del difensore lungo il confine, quindi disegnano il difensore sullo schermo in una nuova posizione.
100-110	Procedura attivata al termine del gioco.
120-160	Stampano il messaggio finale, il punteggio ed il punteggio record.
170-190	Offrono l'opportunità di riprovare e saltano in punti opportuni del programma in base alla risposta.
200-280	Mostrano le istruzioni sullo schermo.
290	Fine.

Programma 1. Pallina rimbalzante

```
5 REM  POKE 53281,1:POKE 53280,12
10 PRINT"{CLR}{BLU}"
20 PRINT " PREMETE {GRN}RETURN{BLU} PER FE
   RMARE LA PALLINA"
30 PRINT "{ 3 GIU'}{GRN}[< 40 X>]"
40 I=1:J=14:DI=1:DJ=1
50 POKE 1024+I+40*J,81
60 POKE 55296+I+40*J,2
70 POKE 1024+I+40*J,32
80 I=I+DI:IF I=0 OR I=39 THEN DI=-DI
90 J=J+DJ:IF J=6 OR J=24 THEN DJ=-DJ
110 GET A$:IF A$=""THEN 50
120 IF ASC(A$)<>13 THEN 50
130 PRINT "{CLR}{BLU}"
140 END
```

Programma 2. Difenditi!

```
1 TS=0:O=54272
2 DEF FNR(X)=1144+40*(INT(RND(0)*20)):GOTO
  200
3 IFA$=CHR$(17)THENPOKEN,32:N=N+40
10 PRINT"{CLR}":POKE53281,12:N=1464:SC=0:D
   =5
15 PRINT"{HOME}[<5>]{RVS}{ 40 SPAZI}{OFF}"
   :PRINT"{HOME}{WHT}PUNTI =";SC
20 POKEN,90
22 I=FNR(X):J=FNR(X):K=FNR(X):H=FNR(X)
24 IFH=IORH=JORH=KORI=JORJ=K THEN 22
25 POKEH,42:POKEI,42:POKEJ,42:POKEK,42
30 GETA$:IF A$=CHR$(136)OR A$=CHR$(32) THE
   N 60
32 IFA$=CHR$(94)THEN POKEN,32:N=N-40:GOTO9
   0
34 IFA$=CHR$(17)THENPOKEN,32:N=N+40:GOTO90
35 L=L+1:IFL<D THEN30
36 H=H-1:I=I-1:J=J-1:K=K-1:L=0
37 IF(H-1024)/40=INT((H-1024)/40)THEN100
38 IF(I-1024)/40=INT((I-1024)/40)THEN100
40 IF(J-1024)/40=INT((J-1024)/40)THEN100
42 IF(K-1024)/40=INT((K-1024)/40)THEN100
44 POKE H+1,32:POKEI+1,32:POKEJ+1,32:POKEK
   +1,32:POKEH,42:POKEI,42:POKEJ,42
45 POKEK,42:POKEH+O,2:POKEI+O,2:POKEJ+O,2:
   POKEK+O,2
50 GOTO30
60 FORM=200TO220:POKEN+O,1:POKEN+O,2:NEXT
62 IFH>N AND H<N+40 THEN 72
64 IFI>N AND I<N+40 THEN 74
66 IFJ>N AND J<N+40 THEN 76
68 IFK>N AND K<N+40 THEN 78
70 SC=SC-5:GOTO82
72 POKEH,102:B=H:H=FNR(X):GOTO80
74 POKEI,102:B=I:I=FNR(X):GOTO80
76 POKEJ,102:B=J:J=FNR(X):GOTO80
78 POKEK,102:B=K:K=FNR(X)
```

```
80 SC=SC+10:POKEB,32
82 PRINT"{HOME}[<5>]{RVS}{ 40 SPAZI}{OFF}"
   :PRINT"{HOME}{WHT}PUNTI =";SC
83 IF SC>500 THEN D=0
84 GOTO30
90 IF N<1104 THEN N=1104
92 IF N>1984 THEN N=1984
94 POKEN,90:POKEN+O,0:GOTO30
100 FORC=55377 TO 56257STEP40:POKEC,2:NEXT
    C:FORC=1 TO 100:NEXTC
110 FORC=55377 TO 55327STEP40:POKEC,1:NEXT
    C
120 PRINT"{WHT}GAME OVER"
130 FORC=1 TO 1000:NEXT:POKE53281,0:POKE53
    280,14
140 PRINT"{CLR}{YEL}{ 2 GIU'}IL TUO PUNTEG
    GIO FINALE E':":PRINT"{CYN}";SC:PRINT"
    {YEL}{ 2 GIU'}"
150 IF SC>TS THEN TS=SC
160 PRINT"RECORD = ";TS
170 PRINT"{WHT}{ 3 GIU'}VUOI RIPROVARCI (S
    /N)"
180 GETA$:IF A$="S" THEN 10
185 IF A$="N" THEN END
190 GOTO 180
200 POKE53281,12:PRINT"{CLR}{BLK}":PRINTTA
    B(5);"** DIFENDITI ! **{ 2 GIU'}"
210 PRINTTAB(9);"DI REGENA"
220 PRINT"{ 2 GIU'}PREMI ↑ PER SALIRE":PRI
    NT"PREMI CRSR DOWN PER SCENDERE"
230 PRINT"{GIU'}PREMI F7 O SPAZIO":PRINT"P
    ER FARE FUOCO.{ 3 GIU'}"
240 PRINT"UCCIDI GLI INVASORI"
250 PRINT"{ 2 GIU'}{WHT}PREMI RETURN PER P
    ARTIRE";
260 GETA$:IF A$="" THEN 260
270 IF ASC(A$)=13 THEN 10
280 GOTO260
290 END
```

Capitolo uno

Grafica ad alta risoluzione semplificata

Paul F. Schatz

Una delle opzioni più complicate offerte dal Commodore 64 è il formato ad alta risoluzione, che suddivide lo schermo in 64000 punti, o «pixel». Attivando e disattivando questi pixel potete creare disegni e grafici ricchi di particolari. Ma dato che il linguaggio BASIC manca di comandi grafici specifici, solo i programmatori più evoluti erano in grado di utilizzare questo formato. Questo articolo rappresenta un notevole passo avanti, poiché mostra come si possano aggiungere semplici comandi grafici al BASIC comunemente usato.

Benché le possilità di realizzare grafica ad alta risoluzione, offerte dal Commodore 64, siano rilevanti, l'accesso alla mappa di bit ad alta risoluzione (320 per 200 pixel) ed i comandi BASIC necessari per disegnare su di essa sono inefficienti e scomodi.

Innanzi tutto i sottoprogrammi BASIC di calcolo ed attivazione di uno specifico bit possono essere fonte di confusione e sconcerto, specialmente per i programmatori alle prime armi, poiché i sottoprogrammi richiedono comandi PEEK, POKE, AND e OR. In secondo luogo questi sottoprogrammi sono lenti; parecchi comandi BASIC devono essere interpretati ed eseguiti per tracciare un solo punto. Inoltre, la mappa di bit deve essere ospitata da un'area di memoria normalmente dedicata ai programmi BASIC. Lo spazio disponibile per i programmi BASIC viene quindi limitato dal fatto che le aree di memoria sono spezzate ed alcune di esse sono inutilizzabili per i programmi suddetti.

Una soluzione alle carenze sopra elencate consiste nell'aggiungere al BASIC nuovi comandi, che controllino la grafica ad alta risoluzione. Questo articolo descrive un sistema che permette di aggiungere quattro comandi.

Capitolo uno

Come modificare il BASIC

Dal momento che abbiamo a disposizione dello spazio nella memoria ad accesso casuale (Random Access Memory = RAM) al di sotto della memoria di sola lettura (Read Only Memory = ROM), possiamo fare una copia del BASIC in RAM e quindi modificarla per i nostri scopi. Abbiamo modificato il BASIC inserendo quattro nuovi comandi, SCREEN, HUE, WIPE e PLOT in sostituzione di altri quattro raramente utilizzati, LET, WAIT, CONT e VERIFY.

Ecco, brevemente, come i nuovi comandi sono stati aggiunti al BASIC. Innanzi tutto notate come i nuovi comandi hanno la stessa lunghezza di quelli che essi rimpiazzano. Un nuovo comando deve essere esattamente inserito nello spazio occupato dal vecchio nella tavola di ricerca dei comandi. Quindi, i puntatori ai vecchi sottoprogrammi BASIC vengono cambiati, in modo da identificare i sottoprogrammi di servizio per i nuovi comandi. Infine il sottoprogramma che gestisce i messaggio di errore è stato modificato, in modo che il calcolatore ritorni automaticamente al normale formato di caratteri, se incontra un errore nel corso della esecuzione di un programma.

Una nota per i programmatori

I sottoprogrammi grafici sono stati sviluppati tenendo d'occhio lo spazio tolto alla memoria destinata ai programmi BASIC. Non è stato sprecato un solo byte. Si è ottenuto ciò usando la memoria RAM al di sotto della ROM Kernel per la mappa di bit. L'inserimento di punti sulla mappa suddetta, in questa locazione può essere realizzato in maniera opportuna solo usando sottoprogrammi in linguaggio macchina, dal momento che si devono disattivare le interruzioni e la ROM Kernel per operare sulla memoria RAM tramite comandi di PEEK. La matrice di schermo, usata per il colore di sfondo e di primo piano, è posizionata alla locazione $C000 ed i sottoprogrammi di grafica in linguaggio macchina si estendono dalla locazione $C400 alla $C545.

I nuovi comandi

I quattro nuovi comandi, SCREEN, HUE, WIPE e PLOT vengono

illustrati di seguito.

.SCREEN ‹numero›

Questo comando attiva e disattiva la mappa di bit ad alta risoluzione. Se il numero è 1, viene mostrata la mappa di bit. Se il numero è 0, viene mostrato lo schermo nel normale formato carattere. Qualsiasi altro valore numerico all'infuori di 0 e 1 darà una segnalazione ILLEGAL QUANTITY ERROR.

.HUE ‹numero›, ‹numero›

Questo comando fissa i colori che vengono mostrati sulla mappa di bit. Il primo numero definisce il colore in primo piano (il colore dei bit attivati). Il secondo numero definisce il colore di sfondo. Un numero maggiore o uguale a 16 darà una segnalazione di ILLEGAL QUANTITY ERROR. I codici colore sono:

0	Nero	4	Porpora	8	Arancio	12	Grigio 2
1	Bianco	5	Verde	9	Marrone	13	Verde-chiaro
2	Rosso	6	Blu	10	Rosso chiaro	14	Azzurro
3	Blu-verde	7	Giallo	11	Grigio 1	15	Grigio 3

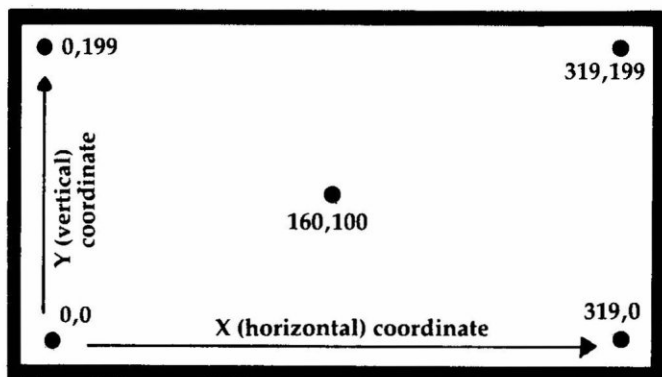
.WIPE

Questo comando provoca una cancellazione ad alta velocità della mappa di bit. Tutti i bit vengono disattivati e lo schermo viene cancellato.

.PLOT ‹numero›, ‹numero›

Questo comando attiva un bit sulla mappa, facendo sì che il pixel corrispondente sullo schermo venga mostrato nel colore di primo piano. Viene usato un sistema di coordinate con origine (0,0) nell'angolo inferiore sinistro (vedi Figura). Il primo numero è l'ascissa del punto ed il secondo ne è l'ordinata. Il primo numero può assumere valori compresi tra 0 e 319, il secondo tra 0 e 199. Numeri al di fuori di questo campo di variabilità provocano un ILLEGAL QUANTITY ERROR.

Capitolo uno

Coordinate usate per tracciare i punti

PLOT X,Y

Come si carica in memoria il nuovo BASIC

Il nuovo BASIC si carica in memoria copiando e mandando in esecuzione il programma 1. Nel copiare il programma siate particolarmente attenti, poiché un numero sbagliato può provocare il blocco totale del calcolatore (costringendovi a spegnerlo per cancellare il contenuto della memoria ROM). Per sicurezza, registrate il programma prima di mandarlo in esecuzione la prima volta. Viene fornito un valore di controllo della somma per avvertirvi della eventuale presenza di un errore nelle istruzioni DATA. Il calcolatore impiega uno o due minuti per eseguire il programma. Per attivare il nuovo BASIC battere in modo diretto:

```
POKE 1,54
```

Il nuovo BASIC può essere disattivato premendo i tasti RUN/STOP e RESTORE contemporaneamente, caricando un altro programma oppure battendo:

```
POKE 1,55
```

Quando si inseriscono programmi che fanno uso dei nuovi comandi grafici, il nuovo BASIC deve essere attivato in modo che il

sottoprogramma di identificazione li possa riconoscere. I comandi che essi rimpiazzano non saranno più funzionanti, a meno che il nuovo BASIC non venga disattivato.

Alcuni semplici programmi

Siamo ora pronti per copiare e mandare in esecuzione un paio di semplici programmi che utilizzano il nuovo BASIC. Il primo è una semplice curva sinusoidale. Caricate e mandate in esecuzione il nuovo BASIC, battete NEW, attivatelo e copiate il programma 2.

Ora battete RUN e vedrete apparire la curva sinusoidale. Non è stato facile?

Ora, cosa ne direste di disporre di una tavola grafica controllata tramite joystick? Assicuratevi di aver registrato il programma 2. Quindi battete NEW e copiate il programma 3. Collegate il joystick alla Port 2 ed usatelo per disegnare sullo schermo. Premete SHIFT-CLR/HOME per cancellare lo schermo o f7 per interrompere il programma.

Questo è solo l'inizio

I programmi scritti nel nuovo BASIC possono essere registrati e letti nel solito modo (ma ricordatevi che abbiamo rinunciato al comando VERIFY). Il nostro scopo era di fornire un utile, seppur rudimentale, strumento per realizzare grafica ad alta risoluzione e dimostrare la facilità con cui il BASIC può essere modificato per includervi nuovi comandi. Esistono numerosi ampliamenti di entrambi gli aspetti che possono venire implementati. Ad esempio, un comando per tracciare linee ad alta velocità, LINE; oppure un nuovo comando simile alla istruzione ON-GOTO, ma con entità del salto dovuta alla posizione del joystick, cioè JOYGOTO, o JOYGOSUB...

Programma 1. Nuovo BASIC

```
0 REM BASIC IN ALTA RISOLUZIONE
10 A=0:REM INIZIALIZZAZIONE CHECKSUM
20 REM SPOSTA IL BASIC DALLA ROM ALLA RAM
30 FORI=40960TO49151:POKEI,PEEK(I):NEXTI
40 REM CAMBIA LET IN HUE
```

```
50 FORI=41150TO41152:READN:POKEI,N:A=A+N:N
   EXTI
60 READL,H:POKE40988,L:POKE40989,H:A=A+L+H

70 DATA 72, 85, 197, 75, 196
80 REM CAMBIA WAIT IN PLOT
90 FOR I=41189TO41192:READN:POKEI,N:A=A+N:
   NEXTI
100 READL,H:POKE41008,L:POKE41009,H:A=A+L+
    H
110 DATA 80, 76, 79, 212, 130, 196
120 REM CAMBIA CONT IN WIPE
130 FORI=41225TO41228:READN:POKEI,N:A=A+N:
    NEXTI
140 READL,H:POKE41024,L:POKE41025,H:A=A+L+
    H
150 DATA 87, 73, 80, 197, 53, 196
160 REM CAMBIA VERIFY IN SCREEN
170 FORI=41201TO41206:READN:POKEI,N:A=A+N:
    NEXTI
180 READL,H:POKE41014,L:POKE41015,H:A=A+L+
    H
190 DATA 83,67,82,69,69,206,11,196
200 REM CAMBIA LA ROUTINE DI MESSAGGI DI E
    RRORE
210 FORI=42042TO42044:READN:POKEI,N:A=A+N:
    NEXTI
220 DATA 76, 0, 196
230 REM LETTURA NUOVE ROUTINE
240 FORI=50176TO50480:READN:POKEI,N:A=A+N:
    NEXTI
250 IFA<>39040THENPRINT"ERRORE NELLE FRASI
     DATA "
260 END
300 DATA 32, 24,196,138, 10,170, 76, 61,16
    4, 80, 70, 83, 32,158,183,224, 1
310 DATA144, 5,240, 19, 76, 72,178,169, 27
    ,141, 17,208,169, 21,141, 24,208
320 DATA169,151,141, 0,221, 96,169, 59,141
```

```
    , 17,208,169, 8,141, 24,208,169
330 DATA148,208,238,162, 32,169,224,133,25
    2,160, 0,132,251,152,145,251,200
340 DATA208,251,230,252,202,208,246, 96, 3
    2,123,196,138, 10, 10, 10, 10,133
350 DATA 2, 32,253,174, 32,123,196,138, 5,
     2,160,192,132,252,160, 0,132
360 DATA251,162, 2,145,251,200,208,251,230
    ,252,202, 16,246,145,251,200,192
370 DATA232,144,249, 96, 32,158,183,224, 1
    6,176, 17, 96, 32,235,183,134, 2
380 DATA169,199, 56,229, 2,133, 2,201,200,
    144, 3, 76, 72,178,165, 21,240
390 DATA 10,201, 1,208,245,165, 20,201, 64
    ,176,239,169, 0,133,251,169,224
400 DATA133,252,165, 20, 41,248, 24,101,25
    1,133,251,165, 21,101,252,133,252
410 DATA165, 2, 41, 7, 24,101,251,133,251,
    144, 2,230,252,165, 2, 74, 74
420 DATA 74, 10,170,189,247,196, 24,101,25
    1,133,251,189,248,196,101,252,133
430 DATA252,165, 20, 41, 7,170,160, 0,120,
    169, 52,133, 1,177,251, 29, 41
440 DATA197,145,251,169, 54,133, 1, 88, 96
    , 0, 0, 64, 1,128, 2,192, 3
450 DATA 0, 5, 64, 6,128, 7,192, 8, 0, 10,
     64, 11,128, 12,192, 13, 0
460 DATA 15, 64, 16,128, 17,192, 18, 0, 20
    , 64, 21,128, 22,192, 23, 0, 25
470 DATA 64, 26,128, 27,192, 28, 0, 30,128
    , 64, 32, 16, 8, 4, 2, 1
```

Programma 2. Una semplice curva sinusoidale

```
10 SCREEN 1: REM ATTIVA LA MAPPA DI BIT
20 WIPE: REM CANCELLA LA MAPPA DI BIT
30 HUE 0,1: REM PUNTI NERI, SCHERMO BIANCO
```

```
40 FOR X=0 TO 319 STEP .5
50 Y=INT(90+80*SIN(X/10))
60 PLOT X,Y: REM TRACCIA UN PUNTO
70 NEXT X
80 GET A$: IF A$="" THEN 80: REM ATTENDE C
   HE VENGA PREMUTO UN TASTO
90 SCREEN 0: REM SCHERMO IN FORMATO NORMAL
   E
```

Programma 3. Tavola di disegno

```
10 SCREEN 1 : WIPE : HUE 0,1
20 X=159: Y=99: PLOT X,Y
30 GOSUB 100: IF J=15 THEN 30
40 PLOT X,Y : GOTO 30
50 SCREEN 0 : END: REM USCITA
100 REM LEGGE IL SEGNALE PROVENIENTE DAL J
    OYSTICK
110 J=PEEK(56320) AND 15: REM PORT 2
120 IF (J AND 8)=0 THEN X=X+1: REM SI MUOV
    E VERSO DESTRA
130 IF (J AND 4)=0 THEN X=X-1: REM SI MUOV
    E VERSO SINISTRA
140 IF (J AND 2)=0 THEN Y=Y-1: REM SI MUOV
    E VERSO IL BASSO
150 IF (J AND 1)=0 THEN Y=Y+1: REM SI MUOV
    E VERSO L'ALTO
160 IF Y<0 THEN Y=0: REM RIMANE NEL CAMPO
    VISIBILE
170 IF Y>199 THEN Y=199
180 IF X>319 THEN X=319
190 IF X<0 THEN X=0
200 GET A$:IF A$=CHR$(147) THEN WIPE: REM
    CANCELLA LO SCHERMO
210 IF A$=CHR$(136) THEN 50: REM TASTO F7
    PER USCIRE
220 RETURN
```

Capitolo due

Formati grafici

La memoria dedicata alla grafica

Sheldon Leemon

Capire com'è organizzata e come viene utilizzata la memoria del Commodore 64 è essenziale per capire qual è la posizione più opportuna in cui sistemare i dati necessari per realizzare la grafica.

I calcolatori Commodore hanno fatto molta strada dai tempi del PET, quando l'argomento della memoria riservata alla grafica poteva essere completamente trattato dicendo che la memoria di schermo era localizzata a partire dalla cella 32768. Il Commodore 64 mette a disposizione grafica a mappa di bit, caratteri grafici e sprite, grazie al chip VIC-II, un sofisticato strumento per visualizzare grafici, che si prende cura di tutti i dettagli connessi con l'organizzazione di ciò che appare sullo schermo. Per poter utilizzare uno qualsiasi di questi formati grafici, tuttavia, il chip VIC-II deve cercare tra i dati contenuti nella memoria per stabilire che cosa visualizzare. Quindi, per quegli utenti che vogliono sfruttare a fondo le potenzialità grafiche del C64 il problema di dove sistemare questi dati in memoria è rilevante.

Potreste pensare che con 64 Kbyte di RAM non ci dovrebbero essere problemi per trovare uno spazio adeguato in cui piazzare la memoria dedicata alla grafica. Ma il chip VIC-II può indirizzare solo 16 Kbyte di memoria contemporaneamente. All'interno di quest'area i dati per gli sprite possono essere posti in uno qualsiasi dei 256 gruppi di 64 byte ciascuno. I dati per i caratteri possono essere registrati in uno degli otto blocchi di 2Kbyte. La memoria dello schermo in formato testo può essere in una delle 16 aree da 1 Kbyte e la memoria per lo schermo in formato a mappa di bit in una delle due sezioni da 8 Kbyte.

Quando accendete il calcolatore il chip VIC-II usa i 16 Kbyte

inferiori di memoria per la grafica. Sfortunatamente, questo blocco di memoria è anche ampiamente usato per altri importanti scopi. Le prime 1024 locazioni sono riservate come spazio di memoria di lavoro RAM per il sistema operativo. Le seconde 1024 locazioni vengono occupate dalla memoria di schermo. Il testo dei programmi BASIC inizia proprio sopra questa locazione. Non è necessario precisare che non rimane molto spazio per gli sprite, i caratteri e gli schermi da 8 Kbyte della bitmap. Benché ci siano dei modi che consentono di eliminare alcuni di questi conflitti, come vedremo più avanti, queste soluzioni sono ben lungi dall'essere esaurienti. In parecchie occasioni un po' più di flessibilità sarebbe utile.

Flessibilità

Fortunatamente, il C64 ha un certo grado di flessibilità. Anche se il chip VIC-II può indirizzare solo 16 Kbyte di memoria contemporaneamente, potete controllare quale blocco di 16 Kbyte utilizzare. Questa possibilità di selezionare il banco viene utilizzata maneggiando i bit 0 e 1 della Port A del secondo chip CIA. Tutto ciò può sembrare un po' complicato, ma in realtà comporta una semplice istruzione POKE. Questi bit devono essere posti come uscite per cambiare i banchi di memoria (questa è la condizione standard al momento dell'accensione). La tecnica richiesta per operare questo cambiamento da programma BASIC verrà discussa più avanti. Ma prima di proseguire e di cominciare a cambiare i banchi di memoria, esaminiamoli uno per uno per vedere quali aree sono disponibili per ogni formato grafico.

Banco 0 (0-16383) [$0000-$3FFF]

Quest'area viene normalmente usata per le variabili di sistema e per il testo dei programmi BASIC. Le locazioni 1024-2048 ($400-$800) sono riservate alla posizione standard della memoria di schermo.

Esiste una ulteriore limitazione, che si applica all'utilizzo come, memoria di questo blocco e del blocco 2. Tutti i dati che il chip VIC-II «vede» devono essere inclusi nello stesso blocco di 16 Kbyte, compresi i dati contenuti nella ROM generatrice di caratteri, che dice al chip come tracciare la forma di ogni lettera sullo schermo. Dal momento che questa ROM non può essere inserita proprio al centro dell'area riservata al testo dei programmi BASIC, per il suo indirizzamento si utilizza un espediente. Come risultato di questo,

espediente il chip VIC-II «vede» la ROM generatrice di caratteri in 4096-8191 ($1000-$1FFF), anche se il microprocessore 6510 la indirizza in 53248 ($D000). Quindi, mentre il 6510 usa la RAM a queste locazioni per il testo del programma, il VIC-II vede la sola ROM e non presta attenzione al contenuto della RAM a queste locazioni. Questa parte di memoria non è quindi disponibile per contenere l'aspetto degli sprite, i caratteri ridefiniti dall'utente o la memoria di schermo, sia essa in formato alta risoluzione o testo.

Come precisato più sopra, rimane ben poco spazio in questo blocco per i dati necessari alla visualizzazione della grafica. Le locazioni 679-767 sono inutilizzate e possono contenere la forma di uno sprite (numero 11) oppure i dati per 11 caratteri. L'area 820-1023 ($334-$3FF), che comprende il buffer del registratore, è disponibile per la memoria dedicata alla grafica, ed è abbastanza grande per contenere la forma di 3 sprite (numeri 13, 14 e 15) o i dati per 25 caratteri. Ma la stessa cosa nei disegni a mappa di bit, che richiede 8 Kbyte di memoria per la visualizzazione su schermo, è un po' più difficile da inserire.

Una soluzione consiste nell'utilizzare parte dell'area normalmente impiegata dal testo del programma BASIC. Si può realizzare ciò sia abbassando il puntatore alla estremità superiore dell'area riservata al testo BASIC, proteggendo così la parte superiore della memoria dalla collisione con il BASIC, oppure innalzando i puntatori all'inizio del testo BASIC, in modo da proteggere la memoria al di sotto di quel punto. Per abbassare l'estremità superiore della memoria BASIC è sufficiente cambiare il puntatore di sistema alla sommità della memoria. Ad esempio, potete riservare la memoria da 8192 in poi con una istruzione POKE 56,32:CLR. Questa istruzione sposta la sommità del BASIC in 32*256, cioè 8192, inserendo questo valore, tramite una istruzione POKE, nel byte più significativo del puntatore. Lo spazio da 8192 a 16384 può essere così usato per lo schermo ad alta risoluzione, un nuovo set di caratteri, la forma degli sprite od uno schermo in formato testo alternativo. Naturalmente, una simile soluzione limita fortemente lo spazio totale restante per i programmi BASIC.

L'alternativa consiste nell'innalzare l'inizio del testo BASIC. Ad esempio, se aveste voluto piazzare uno schermo a mappa di bit di 8 Kbyte alla locazione 8192, avreste potuto spostare l'inizio del BASIC in 16384, per proteggere quell'area di memoria, il che vi avrebbe lasciato con 24Kbyte per un programma BASIC. In modo

immediato, battete la riga seguente:

```
POKE 44,64:POKE 16384,0:NEW
```

Questa soluzione presenta l'inconveniente di essere un po' complicata da implementare senza cambiare il puntatore in modo immediato prima di caricare e mandare in esecuzione il programma.

Banco 1 (16384-32767) [$4000-$7FFF]

Questa sezione viene normalmente utilizzata per ospitare un programma BASIC. Quando si usa questo banco, il chip VIC-II non ha accesso alla ROM generatrice di caratteri.

Una volta che vi siate premurati di abbassare la sommità della memoria, in modo che i programmi BASIC non vi interferiscano, quest'area è a completa disposizione per contenere la forma degli sprite, i caratteri grafici e la grafica a mappa di bit. Gli inconvenienti derivanti dall'uso di questo banco di memoria sono rappresentati dall'indisponibilità della ROM generatrice di caratteri e dalla limitazione dello spazio disponibile per il BASIC (solo 14 Kbyte). L'assenza della ROM carattere è un inconveniente di importanza relativamente minore, poiché potete sempre attivarla e copiare qualsiasi carattere desideriate, o anche tutti, nella RAM. Nonostante il fatto che il problema delle dimensioni possa essere in qualche modo alleggerito, se ci si limita ad occupare la parte superiore di questo banco di memoria, rende comunque questa scelta meno elegante per qualsiasi impiego che non sia quello concernente la grafica a mappa di bit.

Dato che questo blocco è l'unico interamente compreso nella memoria RAM disponibile, esso rappresenta una scelta relativamente buona per quanto riguarda la grafica a mappa di bit. Usando i 9 Kbyte superiori per lo schermo a mappa di bit e per la mappa dei colori, vi rimarrà ancora uno spazio di 21Kbyte per i programmi. La mancanza della ROM carattere non è importante nel formato a mappa di bit, anzi rappresenta in realtà un vantaggio, poiché vi permette di usare entrambe le sezioni di 8 Kbyte.

Banco 2 (32768-49151) [$8000-$BFFF]

Questo blocco consiste di 8 Kbytek RAM, metà della quale viene vista dal chip VIC-II come ROM carattere, ed 8 Kbyte di ROM interprete BASIC.

L'area della ROM BASIC non è, come potreste pensare, total-

mente indisponibile per la grafica. A causa del suo particolare sistema di indirizzamento, a parte la ROM carattere, il chip VIC-II legge solo dalla RAM. Ed anche se il chip del microprocessore 6510 non è in grado di leggere questa RAM, finché la ROM BASIC è attivata (una istruzione di PEEK mostrerebbe solo il contenuto della ROM), può scrivervi (con una istruzione POKE, ad esempio). Qualunque cosa venga scritta nella RAM alla base della ROM BASIC viene mostrata normalmente dal chip VIC-II. Ciò libera un'area extra di 8 Kbyte, per sprite e dati dei caratteri, al di sotto della ROM BASIC.

Dovreste tener ben a mente che mentre potete scrivere in quest'area, non potete leggervi da programma BASIC. Ciò può anche non rappresentare un grave problema per quanto riguarda il set di caratteri e i dati degli sprite, ma è più di un semplice inconveniente, se desiderate usare questa RAM come memoria di schermo. Ad esempio, il sistema operativo deve leggere lo schermo in formato testo per spostare in maniera opportuna il cursore, e se legge il valore contenuto nella ROM invece che i dati dello schermo in RAM, viene indotto in errore in maniera irreparabile, rendendo impossibile battere qualsiasi comando. Similmente, non potreste leggere lo schermo ad alta risoluzione, se fosse piazzato qui, senza far uso di qualche espediente in linguaggio macchina. Con le locazioni 36864-40959 occupate dalla ROM carattere rimangono solo 4 Kbyte di RAM libera per uso come memoria di schermo, insufficienti per uno schermo completo ad alta risoluzione. Quindi, questo blocco non è raccomandabile per l'uso nel formato a mappa di bit, se è necessario che il vostro programma controlli lo schermo. Altrimenti questo è un punto abbastanza buono come memoria grafica, particolarmente se dovete simulare la configurazione di schermo del PET.

Banco 3 (49152-65535) [$C000-$FFFF]

Questo blocco contiene normalmente una RAM di 4 Kbyte completamente inutilizzata dal sistema operativo, un registro di ingresso/uscita di 4 Kbyte e la ROM Kernal da 8 Kbyte del Sistema Operativo. È molto conveniente usarlo quando avete bisogno di molto spazio in memoria per la grafica. Innanzi tutto è ben al di fuori dell'area riservata ai programmi BASIC, quindi non è necessario che cambiate dei puntatori per proteggere i dati grafici dal BASIC e non

dovete limitare lo spazio disponibile per il vostro programma. In pratica, dal momento che non avrete bisogno dell'area di memoria da 1024 a 2048 come memoria di schermo, se usate questo blocco, potete abbassare il puntatore al testo del programma BASIC ed ottenere, se necessario, un altro Kbyte di spazio in memoria per il programma BASIC. In secondo luogo questo blocco ha RAM a sufficienza per quattro schermi in formato testo, mentre l'area ROM può essere usata per immagazzinare contemporaneamente due set di caratteri e la forma di 64 sprite. Anche se la ROM carattere non è disponibile, può essere rapidamente copiata negli ultimi 4 Kbyte al di sotto della ROM Kernal con il seguente programma in linguaggio macchina:

Copia ROM

```
10 FORI=1TO33:READA:POKE49151+I,A:NEXT:REM
    INIZIALIZZA LA ROUTINE IN ML
20 POKE56334,PEEK(56334)AND254:REM DISABIL
   ITA LE INTERRUZIONI
30 POKE1,PEEK(1)AND251:REM SPOSTA I CARATT
   ERI ROM ALLA LOCAZIONE DI MEMORIA 6510
40 SYS49152:REM COPIA IL SET CARATTERI ROM
    NELLA RAM IN LOCAZIONE 61440
50 POKE1,PEEK(1)OR4:REM SPOSTA I CARATTERI
    ROM DALLA LOCAZIONE 6510
60 POKE56334,PEEK(56334)OR1:REM ABILITA LE
    INTERRUZIONI
70 DATA169,0:   REM            LDA #00
80 DATA133,251: REM            STA $FB
90 DATA133,253: REM            STA $FD
100 DATA169,208:REM            LDA #$D0
110 DATA133,252:REM            STA $FB+1
120 DATA169,240:REM            LDA #$F0
130 DATA133,254:REM            STA $FD+1
140 DATA162,16: REM            LDX #16
150 DATA160,0:  REM CICLO      LDY #00
160 DATA177,251:REM CICLO 1 LDA ($FB),Y
170 DATA145,253:REM            STA ($FD),Y
180 DATA136:    REM            DEY
190 DATA208,249:REM            BNE CICLO 1
```

```
200 DATA230,252:REM           INC $FB+1
210 DATA230,254:REM           INC $FD+1
220 DATA202:      REM         DEX
230 DATA208,240:REM           BNE CICLO
240 DATA96:       REM         RTS
```

Anche se questo esempio trasferisce il set di caratteri della ROM in RAM, alla locazione 61440, potete cambiare la destinazione in qualsiasi numero di pagina intero, cambiando l'istruzione DATA di riga 120. Dovete semplicemente sostituire l'indirizzo della nuova destinazione, diviso per 256, al posto del valore numerico 240 (vale a dire 61440/256) fornito da quest'esempio.

Dal momento che non vi è un'area RAM disponibile per lo schermo ad alta risoluzione, è possibile utilizzare l'area al di sotto della ROM Kernal per questo scopo. Benché il contenuto di questa RAM non possa essere letto da BASIC, si può utilizzare un breve programma in linguaggio macchina per disinserire le interruzioni e disattivare la ROM, in modo che si possa usare la RAM. È probabile che la maggior parte dei disegni in formato a mappa di bit venga comunque realizzato in linguaggio macchina, dato che il BASIC risulta troppo lento per poter essere utile a questo scopo.

Un possibile conflitto, di cui voi dovreste essere a conoscenza, è rappresentato dal fatto che la versione corrente del programma di gestione dei dischi DOS (Disk Operative System) è stata scritta per risiedere alla locazione 52224 ($CC00). Sarebbe più sicuro evitare di usare le locazioni da 52224 a 53247 per la memoria grafica, se pensate di usare un programma DOS.

Come cambiare banco di memoria

Ora che abbiamo esaminato i banchi di memoria che si possono utilizzare come memoria riservata alla grafica, ricapitoliamo i passi necessari per operare il cambiamento. Essi sono:

1. Scegliere un banco. Si possono selezionare i banchi da 0 a 3, inserendo le seguenti righe:

```
POKE 56578,PEEK(56578)OR 3:REM SI
PREDISPONE IN USCITA SE NON LO E'
GIA'
POKE 56576,(PEEK(56576)AND 252)OR
BANCO:REM DOVE BANCO E' IL NUMERO
CHE CONTRADDISTINGUE IL BANCO, E
DEVE ESSERE COMPRESO TRA 0 E 3
```

2. Predisporre il registro VIC-II per la memoria carattere. Dal momento che il chip può utilizzare qualsiasi segmento di 2Kbyte all'interno del banco come memoria carattere, dobbiamo inizializzare questo registro per comunicare al chip dove è memorizzata la forma dei caratteri. La formula per fare ciò è:

```
POKE 53272,PEEK(53272)OR TK:REM TK
```

Ad esempio, il set di caratteri della ROM compare nel banco 0 con uno scarto 2 dall'inizio del banco da 4096 byte (4 Kbyte). Quindi, per indirizzare il chip a questo set ROM, dovreste utilizzare l'istruzione POKE 53272, PEEK(53272) OR 4.

Ricordatevi: nei banchi 1 e 3 la ROM carattere non è disponibile, quindi avrete bisogno di spostare il set di caratteri da ROM a RAM, come mostrato nel programma esemplificativo precedente.

3. Inizializzare il registro VIC-II per la visualizzazione della memoria. Dal momento che il chip può utilizzare qualsiasi segmento da 1 Kbyte all'interno del blocco come memoria di schermo, dobbiamo predisporre questo registro per segnalare al chip dove si trovano i dati riguardanti la forma dei caratteri. La formula è la seguente:

```
POKE 53272,PEEK(53272)OR K*16:REM
K E' LO SCARTO DI 1 KBYTE RISPETT
O ALL'INIZIO DEL BLOCCO
```

Ad esempio, nel banco Ø l'area di schermo standard è posta a partire dalla locazione 1024, con uno scarto di 1 Kbyte rispetto all'inizio del blocco. Per predisporre il registro in modo che punti a questa locazione dovreste impiegare una istruzione POKE 53272, PEEK(53272) OR 16.

Dal momento che i passi 2 e 3 eseguono operazioni sullo stesso registro, potreste combinarli ed utilizzare la sola istruzione POKE 53272, (16*K+TK). Usando i valori per difetto dei due esempi precedenti, dovreste utilizzare una istruzione POKE 53272,20.

4. Predisporre il puntatore di sistema per visualizzare la memoria. Anche se avete già segnalato al chip VIC-II dove mostrare la memoria per lo schermo, il sistema operativo (SO) non sa ancora dove deve scrivere i suoi caratteri alfanumerici. Facciamoglielo sapere con questa istruzione:

```
POKE 648,AD/256:REM AD E' L'INDIRI
ZZO CORRENTE DELLA MEMORIA DI SCHE
RMO
```

Noterete come questo puntatore non utilizza uno scarto relativo rispetto all'inizio della memoria del VIC-II, bensì l'indirizzo effettivo della memoria di schermo. Per calcolare questo indirizzo dovrete aggiungere l'indirizzo di base allo scarto. Ad esempio, se lo schermo è posizionato con uno scarto di 1 Kbyte rispetto al terzo banco di memoria, la sua locazione sarebbe 1024+49152, vale a dire 50176. Se dividete questa cifra per 256, troverete che il valore da inserire tramite una istruzione POKE è 196.

Quando avrete eseguito tutte le operazioni elencate non vi sarà alcun cambiamento visibile, a parte forse l'apparizione di un po' di «rifiuti» sullo schermo. Ma se provate ad operare sulla memoria di schermo tramite delle POKE, usando la locazione di partenza standard 1024, non apparirà niente. Potrete effettivamente accorgervi di quanto è successo premendo i tasti STOP e RESTORE. Questa serie di operazioni sposta la locazione standard della memoria di schermo in 1024 nel banco 0, ma il puntatore del SO non è cambiato (per lo meno non nei calcolatori con le prime versioni della Kernal). Come risultato, ciò che voi stavate battendo non verrà mostrato sullo schermo. Esistono due metodi per evitare questo problema. Il sistema più semplice consiste nel disinserire completamente il tasto RESTORE. Con la versione attuale del Sistema Operativo della ROM Kernal dovete solo eseguire una istruzione POKE 792,193 (POKE792,71 ripristina le normali funzioni). Ma se volete che il tasto RESTORE ripristini effettivamente i parametri standard di visualizzazione, dovete aggirare l'Interruzione Non-Mascherabile (INM), che viene provocata dal tasto RESTORE tramite un sottoprogramma in linguaggio macchina che ri-inizializza il puntatore del SO al valore numerico standard 4. Un esempio di questa tecnica è mostrato nel programma 2.

Come si assembla il tutto

Per collegare le operazioni precedenti concluderemo con un

paio di esempi su cambiamenti di banco della memoria di schermo. Il primo vi dimostra come configurare il vostro Commodore 64, in modo che la sua memoria di schermo e il testo dei programmi BASIC inizino negli stessi punti che essi utilizzano sul PET. Il secondo è una dimostrazione più elaborata dell'uso del terzo banco di memoria e comprende un sottoprogramma di trasferimento in linguaggio macchina per spostare il set di caratteri della ROM in RAM ed un breve sottoprogramma di interruzione per ovviare al problema del tasto RESTORE. Dopo l'attivazione viene utilizzato un ciclo per inserire caratteri nella nuova area di memoria dello schermo, tramite istruzioni POKE. Quindi i dati dei caratteri vengono lentamente cancellati, per dimostrare che il set di caratteri risiede ora in RAM. Poi viene usato un ciclo per leggere le locazioni del set di caratteri e scrivere nelle stesse locazioni. Questo dimostra come il 6510 legga la ROM Kernal quando operate tramite istruzioni PEEK su queste locazioni, ma esegua operazioni POKE sulla RAM che viene visualizzata. Infine viene utilizzato di nuovo lo spostamento tramite linguaggio macchina, per mostrare quanto velocemente il set venga ripristinato.

Programma 1. Come configurare il Commodore 64 come un PET

```
10 REM ESEMPIO 1--CONFIGURAZIONE DEL C 64
   COME UN PET
20 POKE53576,PEEK(5676)AND253:REM FASE 1,
   ABILITA BANK 2
30 POKE53272,4:REM FASI 2-3, INDIRIZZA IL
   VIC-II ALLE MEMORIE DI SCHERMO E
40 REM CARATTERE. LO SCARTO PER LO SCHERMO
    E' 0*16, PER IL CARATTERE E' 4
50 POKE648,128:REM FASE 4, INDIRIZZA SO AL
   LO SCHERMO IN 32768 (128*256)
60 POKE44,4:POKE1024,0:REM SPOSTA L'INIZIO
    DEL BASIC IN 1024 (4*256)
70 POKE56,128:CLR:REM LIMITE INFERIORE DEL
   LA MEMORIA IN 32768
80 POKE792,193:REM DISABILITA IL TASTO RES
   TORE
90 PRINTCHR$(147):REM CANCELLA LO SCHERMO
```

Programma 2. Come si usa il Banco 3

```
10 REM ESEMPIO 2, DIMOSTRAZIONE DELL'USO D
   EL TERZO BANCO DI MEMORIA
20 FORI=1TO33:READA:POKE49151+I,A:NEXT:REM
    PREDISPONE IL SOTTOPROGRAMMA IN LM
30 GOSUB200:REM LM COPIA IL SET DI CARATTE
   RI ROM IN RAM
40 POKE56576,PEEK(56576)AND252:REM FASE 1,
    ABILITA IL TERZO BANCO
50 POKE53272,44:REM FASI 2-3, INDIRIZZA IL
    CHIP VIC-II ALLA MEMORIA SCHERMO E
60 REM ALLA MEMORIA CARATTERE. LO SCARTO D
   ELLO SCHERMO E' 2*16, CARATTERE 1212
70 POKE648,200:REM FASE 4, SEGNALA AL SO L
   A POSIZIONE DELLO SCHERMO IN 51200
80 PRINTCHR$(147):REM CANCELLA LO SCHERMO
90 FORI=53236TO53245:READA:POKEI,A:NEXT:RE
   M SOTTOPROGRAMMA INTERRUZIONE NEW
100 POKE53246,PEEK(792):POKE53247,PEEK(793
    ):REM REGISTRA IL VETTORE NMI
110 POKE792,244:POKE793,207:REM FA SERVIRE
     LE INTERRUZIONI DALLA NUOVA ROUTINE
120 FORI=0TO255:POKE51400+I,I:POKE55496+I,
    1:NEXT
125 REM INSERISCE CARATTERI SULLO SCHERMO
    TRAMITE ISTRUZIONI POKE
130 FORJ=1TO8:FORI=61439+JTOI+2048STEP8
140 POKEI,0:NEXTI,J:REM CANCELLA IL SET DI
     CARATTERI
150 FORI=61440TOI+2048:POKEI,PEEK(I):NEXT:
    REM TRASFERISCE ROM IN RAM CON POKE
160 GOSUB200:END:REM RIPRISTINA IL SET DI
    CARATTERI
200 POKE56334,PEEK(56334)AND254:REM DISABI
    LITA LE INTERRUZIONI
210 POKE1,PEEK(1)AND251:REM TRASFERISCE RO
    M CARATTERE IN MEMORIA IN 6510
220 SYS49152:REM COPIA SET DI CARATTERE RO
    M IN RAM IN 61440
```

```
230 POKE1,PEEK(1)OR4:REM TRASFERISCE ROM C
    ARATTERE DALLA LOCAZIONE 6510 MEMORIA
240 POKE56334,PEEK(56334)OR1:REM ABILITA L
    E INTERRUZIONI
250 RETURN
300 REM DATI PER IL PROGRAMMA IN LINGUAGGI
    O MACCHINA CHE COPIA IL SET DI
305 REM CARATTERI IN RAM.
310 DATA169,0,133,251,133,253,169,208,133,
    252,169,240,133,254,162,16
320 DATA160,0,177,251,145,253,136,208,249,
    230,252,230,254,202,208,240,96
330 REM IL SUCCESSIVO E' UN PROGRAMMA IN L
    M PER FAR SI CHE IL TASTO RESTORE
335 REM RIPRISTINI IL PUNTATORE SO ALLO SC
    HERMO.
340 DATA72,169,4,141,136,02,104,108,254,20
    7
```

Impariamo la grafica «bitmap»

Michael Tinglof

Come orizzontarvi attraverso i bit ed i byte dello schermo ad alta risoluzione grafica, con un sottoprogramma in linguaggio macchina che potete utilizzare nei vostri programmi per tracciare e cancellare punti.

Lo schermo ad alta risoluzione grafica è composto da piccoli punti: 64000, per essere precisi. Ciascuno di essi può essere acceso o spento. Dal momento che ciascun punto, o «pixel», può essere controllato individualmente, il vostro calcolatore deve avere una informazione di acceso-spento per ciascuno di essi. Se usaste un byte per ciascun pixel, si richiederebbero suppergiù tutti i byte di RAM disponibili. Non ci sarebbe più spazio per il BASIC o la Kernal od il Sistema Operativo o qualsiasi altra cosa.

Ma non è necessario che ci sia un byte per ciascun punto dello schermo, dato che otto pixel possono essere controllati da un solo byte tramite un metodo chiamato «bitmap», vale a dire «a mappa di bit».

Sembra alta matematica, ma non lo è

Se non conoscete ancora la matematica binaria, non importa. Potete utilizzare il formato a mappa di bit pur senza capire a fondo il complemento a due, lo scorrimento aritmetico verso sinistra e lo scorrimento logico verso destra. Tutto ciò che dovete conoscere è come attivare e disattivare i punti dello schermo.

Ciascun byte consiste di otto bit. Sono come otto interruttori messi in fila. Gli interruttori possono essere accesi o spenti.

Gli otto bit di un byte possono essere sia uguali a 1, il che significa *accesi,* sia a 0, ossia *spenti.* Il chip VIC-II scandisce la memoria leggendo ciascun bit di ogni byte. Se il bit è *acceso,* cioè uguale a 1, il chip VIC attiverà il punto corrispondente dello schermo. Se il bit è

spento, cioè uguale a 0, il chip VIC manterrà quel punto nello stesso colore dello sfondo dello schermo.

La Figura 1 rappresenta una mappa di bit per uno schermo molto piccolo. Questo schermo è alto esattamente 8 pixel e largo 32. Ciascun 1 rappresenta un punto attivato e ciascuno 0 un punto nello stesso colore dello sfondo.

Figura 1. Una mappa di bit 32x8

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0
0 0 0 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 0 0 0
0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0
```

In Figura 2 vengono mostrati i soli bit *attivati,* in modo che possiate vedere come questo schermo in formato ridotto contenga un disegno molto semplificato del profilo di una vettura.

Figura 2. I bit attivati

Ogni gruppo di otto punti è controllato da un singolo byte. Ogni bit del byte controlla un punto. La Figura 3 mostra questa mappa di bit suddivisa nei byte che la compongono. La rappresentazione del valore decimale di ciascun byte compare sotto la rappresentazione binaria dello stesso.

Figura 3. I byte nella mappa di bit

0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
(0)	(0)	(0)	(0)
0 0 0 0 0 0 0 0	0 1 1 1 1 1 1 1	1 1 1 1 1 1 0 0	0 0 0 0 0 0 0 0
(0)	(127)	(252)	(0)
0 0 0 0 0 0 0 0	1 0 0 0 0 0 0 0	0 0 0 0 0 0 1 0	0 0 0 0 0 0 0 0
(0)	(128)	(2)	(0)
0 0 0 1 1 1 1 1	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 1	1 1 1 1 1 0 0 0
(31)	(0)	(1)	(248)
0 0 1 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 1 0 0
(32)	(0)	(0)	(4)
0 1 1 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 1 1 0
(96)	(0)	(0)	(6)
0 0 0 1 1 1 1 0	0 0 1 1 1 1 1 1	1 1 1 1 1 0 0 0	1 1 1 1 1 0 0 0
(30)	(63)	(248)	(248)
0 0 0 0 0 0 0 1	1 1 0 0 0 0 0 0	0 0 0 0 0 1 1 1	0 0 0 0 0 0 0 0
(1)	(192)	(7)	(0)

Ora vediamo com'è organizzata in memoria questa mappa di bit in formato ridotto. La memoria è organizzata come una sola lunga sequenza di byte, dalla locazione 0 alla 65535. Ma il chip VIC-II legge la memoria a mappa di bit come se fosse suddivisa in un grande set di caratteri. Vale a dire, legge la memoria come se fosse divisa in *celle* larghe otto bit ed alte altrettanto. Esistono 1000 di queste *celle,* in formato 40x25. La Figura 4 rappresenta una mappa delle celle di schermo. Ci sono esattamente tante celle nella mappa di bit quanti sono i pixel nella normale memoria di schermo.

Figura 4. Mappa delle celle

	+0	+10	+20	+30
0 +	0123456789	0123456789	0123456789	0123456789
40 +	0123456789	0123456789	0123456789	0123456789
80 +	0123456789	0123456789	0123456789	0123456789
120 +	0123456789	0123456789	0123456789	0123456789
160 +	0123456789	0123456789	0123456789	0123456789
200 +	0123456789	0123456789	0123456789	0123456789
240 +	0123456789	0123456789	0123456789	0123456789
280 +	0123456789	0123456789	0123456789	0123456789
320 +	0123456789	0123456789	0123456789	0123456789
360 +	0123456789	0123456789	0123456789	0123456789
400 +	0123456789	0123456789	0123456789	0123456789
440 +	0123456789	0123456789	0123456789	0123456789
480 +	0123456789	0123456789	0123456789	0123456789
520 +	0123456789	0123456789	0123456789	0123456789
560 +	0123456789	0123456789	0123456789	0123456789
600 +	0123456789	0123456789	0123456789	0123456789
640 +	0123456789	0123456789	0123456789	0123456789
680 +	0123456789	0123456789	0123456789	0123456789
720 +	0123456789	0123456789	0123456789	0123456789
760 +	0123456789	0123456789	0123456789	0123456789
800 +	0123456789	0123456789	0123456789	0123456789
840 +	0123456789	0123456789	0123456789	0123456789
880 +	0123456789	0123456789	0123456789	0123456789
920 +	0123456789	0123456789	0123456789	0123456789
960 +	0123456789	0123456789	0123456789	0123456789

Figura 5. Una cella di 8 byte

byte	configurazione di bit
0	00000000
1	00000000
2	00000000
3	00000000
4	00000000
5	00000000
6	00000000
7	00000000

Ogni cella consiste di otto byte. Questo dà una configurazione di 8x8 bit. Il chip VIC-II legge ciascun byte della cella in sequenza dall'alto in basso prima di proseguire con la cella successiva, come mostrato in Figura 5.

Il cammino generalmente seguito dal chip VIC, quindi, consiste nell'iniziare a leggere la memoria di schermo in cella 0, posta nell'angolo superiore sinistro dello schermo. Gli otto byte di questa cella vengono letti in ordine dall'alto in basso. Quindi il VIC-II legge la cella 1, che si trova nella riga superiore, immediatamente alla destra della cella 0. Il VIC-II prosegue finché non raggiunge l'ultima cella della prima riga, la 39. Quando legge la cella 40 inizia una nuova riga.

Questo significa che seguendo questo percorso la nostra minuscola mappa di bit delle Figure da 1 a 3 apparirebbe *in memoria* come mostrato in Figura 6. Se la mappa di bit iniziasse all'indirizzo 16384, trovereste i byte nell'ordine raffigurato. I primi otto byte rappresentano la cella 0, i successivi otto la cella 1 e così via.

Figura 6. La mappa di bit in memoria

Disposizione dei byte in memoria

Cella 0	**Cella 1**	**Cella 2**	**Cella 3**
0	0	0	0
0	127	252	0
0	128	2	0
31	0	1	248
32	0	0	4
96	0	0	6
30	63	248	248
1	192	7	0

Byte in memoria

Indirizzo	**Byte**	
16384	0	inizio cella 0
16386	0	
16385	0	
16387		

31
16388 32
16389 96
16390 30
16391 1
16392 0 inizio cella 1
16393 127
16394 128
16395 0
16396 0
16397 0
16398 63
16399 192
16400 0 inizio cella 2
16401 252
16402 2
16403 1
16404 0
16405 0
16406 248
16407 7
16408 0 inizio cella 3
16409 0
16410 0
16411 248
16412 4
16413 6
16414 248
16415 0

Operazioni binarie

Come fa il calcolatore a cambiare effettivamente i punti attivati o disattivati? Non potete indirizzare un solo bit per volta in memoria di schermo tramite istruzioni PEEK o POKE; infatti, se volete cambiare un punto sullo schermo, dovete indirizzare l'intero byte tramite una operazione POKE, controllando così otto bit invece di uno solo.

Il C64 fornisce alcuni comandi che vi permettono di ricavare un byte dalla memoria di schermo, cambiare un pixel (o più di uno) individualmente e quindi reinserire il byte nella sua posizione.

Capitolo due

Prima di predisporre un programma che tracci un singolo punto, prepariamo un sottoprogramma che estragga un numero dalla memoria di schermo e quindi lo reinserisca quando abbiamo concluso le nostre operazioni. Supponiamo che la memoria a mappa di bit inizi all'indirizzo MM. Quando si accede a questo sottoprogramma la variabile CW dirà quale cella, da 0 a 999, desideriamo cambiare, e BW dirà quale byte all'interno della cella, da 0 a 7, desideriamo cambiare.

```
500 W=MM+CW*8+BW
510 XB=PEEK(W)
599 POKE W,NB:RETURN
```

La variabile XB contiene il vecchio valore del byte ed NB il valore modificato. W viene inizializzato al valore assoluto dell'indirizzo del byte che ci apprestiamo a modificare: l'indirizzo di partenza della memoria a mappa di bit più il numero della cella (moltiplicato per 8) più il numero del byte all'interno della cella suddetta. In seguito, tra le righe 510 e 599, inseriremo le righe che operano l'effettivo cambiamento del byte.

Ora che abbiamo a disposizione il byte, che cosa ne facciamo?

L'operatore AND. Quando usate una espressione del tipo A = 5 AND 3, la chiave AND provoca l'esecuzione di una *operazione binaria*. I due numeri vengono confrontati bit per bit. Mostriamo le due serie di bit sovrapposti, in modo da facilitare il confronto:

bit:	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
5	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**
3	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**

Notate che i bit sono numerati da destra a sinistra, da 0 a 7. Sembra un po' strano, ma è significativo. Il bit 0 è il bit meno significativo, vale a dire che un 1 in questa posizione vale solo 1. Il bit 1 ha un valore doppio del valore del bit 0: un 1 in questa posizione ha un valore numerico di 2. Il bit 2 ha un valore di 4, il bit 3 ha un valore di 8 e così via. Ecco una lista del valore che un 1 ha in corrispondenza della posizione di ciascun bit:

bit:	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	128	**64**	**32**	**16**	**8**	**4**	**2**	**1**

Ora, quando eseguiamo una operazione AND sui numeri 5 e 3 il

calcolatore confronta ciascun bit del primo numero con il corrispondente bit del secondo numero. Ad esempio, il bit 7 del numero 5 è uguale a 0; il bit 7 del numero 3 è uguale a 0.

Quando l'operatore AND confronta i due numeri cerca un 1 nello stesso bit di entrambi i numeri. Tutte le volte che trova due 1 corrispondenti pone un 1 come risultato dell'operazione; tutte le altre volte pone 0 come risultato:

	bit:	7	6	5	4	3	2	1	0
AND	**5**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**
	3	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**
risultato	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

Nel bit 2 l'operatore AND trova un 1 nel numero 5, ma non vi è un 1 corrispondente nel numero 3. Quindi uno 0 viene inserito nel corrispondente bit del risultato. Nel bit 1 l'operatore AND trova un 1 nel numero 3, ma senza che vi sia un 1 corrispondente nel numero 5 in quella stessa posizione. Risultato? Un altro 0. Solo nel bit 0, dove entrambi i numeri hanno un 1 in quella posizione, il risultato è 1. Quindi 3 AND 5 = 1.

L'operatore OR. L'operatore OR confronta i due numeri cui è applicato nello stesso modo dell'AND, solo che non va a cercare le coppie di 1. Se trova un 1 *in uno qualunque* dei due numeri, pone, in corrispondenza del bit osservato, un 1 nel risultato. Ecco qual è l'aspetto dell'operazione 5 OR 3:

	bit:	7	6	5	4	3	2	1	0
OR	**5**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**
	3	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**
risultato	**7**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**

Dal momento che il numero 5 ha un 1 in corrispondenza del bit 2, ci sarà un 1 nel bit 2 del risultato, indipendentemente da ciò che c'è nel corrispondente bit del numero 3.

La regola, quindi, è:

AND ha come risultato 1 nel caso in cui *entrambi* i numeri presentano un 1 nel bit che viene preso, di volta in volta, in considerazione.

OR ha come risultato 1 nel caso in cui *almeno uno* dei due numeri presenta un 1 nel bit preso in considerazione.

Come si usano gli operatori AND e OR con una mappa di bit. Avrete probabilmente già capito come si possa, in questo modo, attivare o disattivare un singolo pixel. Supponiamo che vogliate attivare il bit 3 del byte, indipendentemente da ciò che vi è già. Tuttavia non volete cambiare uno qualsiasi degli altri bit del byte. Ecco cosa farebbe il nostro sottoprogramma:

```
500 W=MM+CW*8+BW
510 XB=PEEK(W)
520 NB=XB OR 8
599 POKE W,NB:RETURN
```

Che cosa succede in riga 510? In notazione binaria il numero 8 ha questo aspetto: 00001000. Il bit 3 (il quarto bit da destra) è a 1. Tutti gli altri sono zeri. Quando applicate l'operatore OR 8 con qualsiasi numero il valore che ne risulta avrà sempre un 1 in bit 3. OR 8, quindi, attiva il bit 3.

Per attivare qualsiasi bit usate lo stesso procedimento. OR 2 attiva il bit 1. OR 128 attiva il bit 7.

Per attivare due bit limitatevi a sommare le due cifre prima di applicarvi l'operatore OR ed il byte della memoria di schermo. Ad esempio, per attivare i bit 0 ed 1, NB=XB OR (1+2). Per attivare i bit 6 e 7, NB=XB OR (128+64). Naturalmente, non è necessario che eseguiate esplicitamente la somma all'interno del programma. Scriverete l'ultima istruzione nel modo seguente:

```
NB=XB OR 192
```

Come potete attivare tutti i bit di un byte? Applicate l'operatore OR a 1+2+4+8+16+32+64+128, il che dà un totale di 255. NB=XB OR 255. Naturalmente, se avete intenzione di attivare *ciascun* bit, potete limitarvi a porre NB=255. Avete bisogno di utilizzare un

operatore OR solo se volete cambiare alcuni bit all'interno del byte e lasciare gli altri immutati.

L'operatore OR viene utilizzato per *attivare* i bit. AND per *disattivarli.* Per disattivare un punto, ricordatevi, dovete ottenere uno 0 nella posizione voluta.

Qualsiasi numero cui si applichi l'operatore AND 0 darà come risultato 0, dal momento che non vi è possibilità di avere coppie di 1. Quindi, per disattivare tutti i bit di un byte dovete applicargli un operatore AND 0 (tuttavia, se volete cancellare un intero byte, non è necessario utilizzare l'operatore AND: limitatevi ad inserire il valore numerico 0 tramite un'istruzione POKE nella locazione opportuna).

Ma supponiamo di voler cancellare solo il bit 7. Vogliamo lasciare tutti gli altri bit intatti. Dal momento che porre uno 0 nella posizione corrispondente ad un bit lascerà sempre uno 0 in quella stessa posizione per quanto riguarda il risultato, dovete porre un 1 in ogni posizione che volete lasciare intatta. Ecco quale sarebbe l'aspetto del programma:

```
500 W=MM+CW*8+BW
510 XB=PEEK(W)
520 NB=XB AND 127
599 POKE W,NB:RETURN
```

Perché 127? Perché 127=255-128. Prendiamo in considerazione il numero binario:

0 1 1 1 1 1 1 1
(127)

Notate che 127 ha tutti i bit *attivati,* tranne il bit 7. Se applicate l'operatore AND a qualsiasi numero ed a 127, tutti i bit, dal bit 0 al bit 6, che erano eventualmente attivati nel valore numerico originario saranno ancora attivati nel risultato, dal momento che vi è un 1 in 127 per far coppia con essi. Tutti i bit che erano disattivati nel valore numerico originario rimarrebbero disattivati. Con il bit 7, tuttavia, non vi potrebbe essere un possibile accoppiamento, dal momento che vi è uno 0 in quella posizione nel numero 127 ed il risultato sarà sempre 0.

Quindi, per tracciare i punti cominciamo con lo 0, poniamo un 1 in ciascuna posizione che vogliamo attivare e quindi applichiamo l'operatore OR a questo numero ed a quello che già si trova nella memoria di schermo. Per cancellare punti partiamo da 255, poniamo uno 0 in tutte le posizioni che desideriamo disattivare e quindi applichiamo l'operatore AND al numero così ottenuto ed al numero che già si trova nella memoria di schermo.

Per attivare il bit 7 partiamo da 0 e vi aggiungiamo 128, che rappresenta il valore del bit 7 attivato. Quindi, applicare OR 128 al byte di memoria di schermo attiverà il bit 7.

Per disattivare il bit 7 partiamo da 255 e vi sottraiamo 128, il che pone uno 0 nel bit 7, con un risultato di 127. Quindi applichiamo l'operatore OR 127 al byte di memoria di schermo ed il bit 7 verrà disattivato.

Come sistemare la mappa di bit in memoria

Ora che abbiamo visto come funziona la mappa di bit, è tempo di stabilire dove dovrebbe trovarsi in memoria. Per farlo è necessario capire come il chip VIC-II legge la memoria.

Memoria di schermo, memoria colore e mappa di bit. Se avete già avuto a che fare con la grafica realizzata tramite caratteri (l'esatto opposto del formato «bitmap»), sarete probabilmente abituati ad usare sia la memoria di schermo, che consiste di valori simbolici di schermo per i caratteri che devono essere visualizzati su di esso, sia la memoria colore, che è composta dai valori simbolici dei colori per ogni carattere dello schermo.

In formato «bitmap» l'area di memoria colore da 55296 in poi viene ignorata. Tuttavia, i 1000 byte della memoria di schermo vengono ora utilizzati come memoria colore per la mappa di bit. Ciascun byte della memoria di schermo contiene il codice colore per la corrispondente *cella* della mappa di bit. Quindi, d'ora in avanti, ogni volta che parleremo di memoria di schermo intenderemo l'area di memoria in cui viene controllato il *colore,* e quando parleremo di mappa di bit intenderemo l'area di memoria in cui i singoli punti vengono attivati o disattivati.

L'indirizzo di base dell'area grafica. Il VIC-II non può maneggiare 64Kbyte. Può controllare solo 16Kbyte di memoria contemporaneamente, al massimo. Quindi, contrariamente alla CPU 6510, il VIC-II utilizza la memoria come se fosse suddivisa in banchi da

16Kbyte ciascuno, in questo modo:

Banco 0 indirizzi 0-16383
Banco 1 indirizzi 16384-32767
Banco 2 indirizzi 32768-49151
Banco 3 indirizzi 49152-65535

Il VIC-II può leggere uno qualsiasi di questi quattro banchi, ma solo uno per volta. Ciò significa che, se ponete la mappa di bit nel banco 3, allora anche la memoria di schermo deve essere nel banco 3.

Come comunicare al VIC-II quale banco usate? I bit 0 e 1 della locazione 56576 controllano la scelta dei banchi in questo modo:

Banco	**Bit**	**Valore decimale da inserire tramite POKE 56576**
0	11	3
1	10	2
2	01	1
3	00	0

In questo modo una istruzione POKE 56576,0 comunicherà al VIC-II di utilizzare il banco 3, a partire dalla locazione 49152.

Quale blocco conviene usiate per la grafica a mappa di bit? Il migliore è il banco 1, da 16384 a 32767. Perché? Perché gli altri blocchi sono troppo occupati. Il banco 0, il blocco normalmente selezionato dal VIC-II, viene usato anche per i vostri programmi BASIC e contiene parecchie funzioni fondamentali del sistema operativo. I banchi 2 e 3 forniscono parecchio spazio alla ROM. Quindi desidererete probabilmente utilizzare l'istruzione POKE 56576,2.

Il primo indirizzo di ciascun banco è l'indirizzo di base dell'area grafica. Nel calcolo di altri indirizzi userete l'indirizzo di base come punto di partenza e calcolerete le altre locazioni aggiungendo valori numerici all'indirizzo di base (se utilizzate delle variabili per contenere questi indirizzi nel vostro programma, potrete in seguito passare da un banco all'altro cambiando semplicemente il valore numerico contenuto nelle variabili suddette, invece di essere

costretti a cercare ogni volta nel programma il valore che si desidera cambiare).

Il blocco di memoria di schermo. La memoria di schermo (che controlla il colore) occupa quasi 1Kbyte di memoria e la mappa di bit usa circa 8Kbyte. Entrambe devono essere collocate all'interno del banco di 16Kbyte dedicato alla grafica. La memoria di schermo deve iniziare su di un limite di 1Kbyte, vale a dire che il suo indirizzo di partenza deve essere esattamente divisibile per 1024.

Di conseguenza esistono 16 possibili locazioni per la memoria di schermo all'interno del blocco. I seguenti sono i possibili indirizzi di partenza di ogni possibile blocco di memoria di schermo, sotto forma di scarti rispetto all'indirizzo di base dell'area grafica.

Per inserire un valore numerico nell'angolo superiore sinistro della memoria di schermo dovreste utilizzare una istruzione POKE che abbia come argomento l'indirizzo di base dell'area grafica *più* lo scarto rispetto al blocco di memoria di schermo.

Figura 7. Scarti di memoria di schermo possibili

Scarto	Blocco di memoria di schermo	Valore POKE
0	0	0
1024	1	16
2048	2	32
3072	3	48
4096	4	64
5102	5	80
6144	6	96
7168	7	112
8192	8	128
9216	9	144
10240	10	160
11264	11	176
12288	12	192
13312	13	208
14366	14	224
15360	15	240

Per segnalare al VIC-II quale blocco di 1Kbyte state utilizzando come memoria di schermo dovete inserire, tramite istruzione POKE, il numero di blocco *moltiplicato* per 16 nella locazione 53272.

Perché moltiplicato per 16?

Anche la locazione 53272 è sotto forma di mappa di bit! I quattro bit più a sinistra (bit 4-7) della locazione 53272 controllano il blocco della memoria colore. Se i bit 4-7 sono 0000, allora è stato scelto il blocco di memoria colore 0; se sono 0001, è stato selezionato il blocco 1; se hanno la configurazione 0010, è il blocco 2 ad essere selezionato e così via.

Tuttavia, quando state inserendo valori numerici nel byte della locazione 53272 non potete limitarvi ad operare con una istruzione POKE sui quattro bit più significativi: dovete operare sull'intero byte. Ad esempio, se volete scegliere il blocco 7 (numero binario 0111), non potete farlo con il comando POKE 53272,7. Questo comando inserirebbe il numero binario 7, vale a dire 00000111, in quella locazione. I bit da 4 a 7 sarebbero uguali a zero!

Ma se moltiplicate il numero del blocco per 16, ottenete l'effetto di spostare tutti i bit *attivati* di quattro posizioni a sinistra. Invece di compiere una operazione POKE 53272,7 eseguiremo una POKE 53272,7*16, cioè 112. Questo comando inserisce il numero binario 01110000 nella locazione 53272, il che era esattamente ciò che volevamo.

Il blocco a mappa di bit. Dal momento che la mappa di bit utilizza 8Kbyte, esistono solo due possibili blocchi a mappa di bit all'interno del banco di 16Kbyte riservato alla memoria grafica, uno con scarto iniziale 0 ed uno con scarto iniziale 8192, vale a dire 8Kbyte. In altre parole, il blocco a mappa di bit deve occupare o la prima metà o la seconda del banco dedicato alla memoria grafica.

Per segnalare al VIC-II se avete scelto il blocco 0 o il blocco 1 per la mappa di bit, dovete inserire ancora una volta un valore numerico, tramite una istruzione POKE, nella locazione 53272, la stessa locazione dove avete inserito le informazioni relative al blocco di memoria di schermo. Questa volta, però, è il solo bit 3 che sceglie il blocco, quindi per ottenere il valore numerico opportuno dovete moltiplicarlo per 8.

Dal momento che la stessa locazione, 53272, controlla sia la memoria colore che il blocco della mappa di memoria, dovete sommare i due numeri prima di inserirli tramite una istruzione POKE (questo perché inserire con una POKE solo uno dei due numeri provocherebbe l'azzeramento dell'altro). Se la variabile SB contiene il numero del blocco colore (0-15) e la variabile MB il

numero del blocco a mappa di bit (0 o 1), utilizzerete una POKE 53272,SB*16+MB*8.

Figura 8. Scarti possibili per la mappa di bit

Scarto	Blocco della memoria a mappa di bit	Valore POKE
0	0	0
8192	1	8

Quindi, se volete il banco 1 come memoria dedicata alla grafica ed all'interno di questo banco desiderate che il blocco 1 sia riservato alla mappa di bit ed il blocco 7 alla memoria di schermo, ecco ciò che il vostro programma dovrebbe fare:

```
10 GB=2:POKE 56576,GB:REM SCEGLIE
   IL BANCO 1
20 SB=':MB=1:POKE 53272,SB*16+MB*8
   :REM SCEGLIE IL BLOCCO 7 PER LO
   SCHERMO ED IL BLOCCO 1 PER LA M
   APPA DI BIT
30 GM=49152-GB*16384:SM=GM+SB*1024
   :MM=GM+MB*8192:REM INIZIALIZZA
   LE VARIABILI CONTENENTI GLI IND
   IRIZZI
```

In riga 30 questo programma inizializza la variabile GM al valore dell'indirizzo di base della memoria dedicata alla grafica. Quindi inizializza SM al valore dell'indirizzo iniziale della memoria di schermo ed MM all'indirizzo di partenza della memoria a mappa di bit.

Come si attiva la grafica a mappa di bit. Una volta che abbiate inizializzato i puntatori, dovete segnalare al chip VIC-II di passare dalla grafica tramite caratteri grafici alla grafica a mappa di bit. Dovete farlo attivando il bit 5 del byte contenuto nella locazione 53265. Ecco l'istruzione POKE che si occupa di svolgere questo compito:

```
POKE 53265,PEEK(53265) OR 32
```

Capitolo due

Come tracciare punti sullo schermo in formato a mappa di bit

Come si traduce tutto ciò in una normale rappresentazione cartesiana con coordinate X-Y? Ora sapete come utilizzare i comandi AND e OR per inserire dei punti in un byte; sapete come comunicare al calcolatore che volete utilizzare la grafica a mappa di bit e dove può trovare la mappa stessa; ma come fare a segnalare al calcolatore quale bit esattamente desiderate, sull'intero schermo, che sia attivato e quale disattivato?

Per trovare un particolare pixel pensate allo schermo come ad una grande scacchiera quadrettata, con 320 colonne verticali per 250 righe orizzontali. Desiderate riempire un riquadro dello schermo nella posizione X, Y, dove X è il numero della colonna (da 0 a 319) ed Y il numero di riga (da 0 a 249). Se la posizione 0,0 si trova nell'angolo superiore sinistro, questa griglia di coordinate X-Y avrebbe l'aspetto di Figura 9, nella quale il riquadro corrispondente alla coppia di coordinate (3,1) è riempito.

Figura 9. Il sistema di coordinate X-Y

	colonna						
riga	0	1	2	3	4	...	319
0							
1							
2							
3							
4							
.							
.							
249							

Ricordatevi che il punto che stiamo tracciando corrisponde ad un pixel sullo schermo, che viene rappresentato da un singolo bit da qualche parte della mappa di bit. Il punto di coordinate (3,1) sarebbe piuttosto facile da reperire, dal momento che si tratterebbe del terzo bit dalla sinistra (bit 5) nel secondo byte (byte 1) della prima cella (cella 0) della mappa di bit. Ma non sarà sempre così facile.

Ad esempio, il punto di coordinate (299,144) è molto più interno alla mappa di bit. Come possiamo scoprire in quale byte quel bit è contenuto, in modo da poterlo attivare? Il sottoprogramma che abbiamo escogitato in precedenza non svolge questa funzione:

presume che la cella ed il byte siano già stati trovati. Abbiamo bisogno di un programma che possa partire dalla coppia di coordinate di un pixel e trovare il bit nella mappa in base a questa sola informazione.

Ecco un breve programma che lo farà. Prima di questo programma la variabile MM deve essere stata inizializzata all'indirizzo assoluto iniziale della mappa di bit:

```
100 X=299:Y=144
110 XC=INT(X/8)*8:YC=INT(Y/8)*8
120 XB=2^(X-XC):YB=Y-YC
130 PT=MM+YC*320+XC+YB
140 POKE PT,PEEK(PT) OR XB
```

Come funziona questo programma?

Riga	**Commento**
100	Inizializza le variabili X e Y, che contengono le coordinate.
110	Pone XC uguale al numero di colonna moltiplicato per 8; pone YC uguale al numero di riga moltiplicato per 8.
120	Pone YB uguale al numero del byte all'interno della cella. Pone XB uguale al valore decimale del bit da attivare all'interno del byte.
130	Pone PT uguale all'indirizzo assoluto del byte che deve essere tracciato (l'inizio della mappa di bit *più* lo scarto dalla riga della cella *più* lo scarto dalla colonna della cella *più* lo scarto dal byte internamente alla cella).
140	Legge il byte corrente, tramite una istruzione PEEK alla locazione PT, vi applica un operatore OR con argomento XB (il bit da attivare all'interno del byte) e lo reinserisce nella locazione PT tramite una istruzione POKE.

Come si controlla il colore

Nella grafica a mappa di bit il chip VIC-II utilizza la memoria di schermo per stabilire i colori dei pixel *attivati* e *disattivati* della mappa di bit.

Ogni cella della mappa di bit viene controllata, per quanto concerne il colore, da un byte nella memoria di schermo. I quattro bit rimanenti (da 4 a 7) di ogni byte della memoria di schermo con-

trollano il colore che deve essere visualizzato per ogni bit *attivato* nella cella della mappa di bit. I quattro bit di destra (0-3) di ogni byte della memoria di schermo stabiliscono il colore che deve essere visualizzato per i bit *disattivati* della cella della mappa di bit, come mostrato in Figura 10.

Figura 10. Il byte di controllo colore nella memoria di schermo

Colore dei bit attivati	Colore dei bit disattivati
7 6 5 4	3 2 1 0

Questa è una delle opzioni più potenti offerte dalla grafica del Commodore 64. Potete far apparire contemporaneamente fino a 16 colori differenti sullo schermo. Esiste, tuttavia, un inconveniente. Cambiare il colore di un singolo bit farà cambiare il colore di ogni altro bit all'interno della stessa cella. Nonostante ciò, organizzandovi attentamente potete, ottenere disegni ad alta risoluzione, tali da lasciare stupiti, con diversi colori sullo schermo.

Che valori numerici dovete inserire, tramite POKE, nella memoria di schermo? I codici colore vanno da 0 a 15. Per il colore di fondo — il colore che viene visualizzato in corrispondenza di ciascuno 0 nella cella della mappa di bit — dovete solo inserire il codice colore, tramite una POKE, nella memoria di schermo. Per il colore di primo piano — il colore che viene visualizzato in corrispondenza degli 1 della mappa di bit — dovete moltiplicare il codice colore per 16, in modo da spostarlo di quattro bit a sinistra. Se la variabile C1 rappresenta il colore di primo piano e la variabile C0 il colore di fondo, la seguente istruzione porrà il colore voluto nella locazione di memoria di schermo SM:

```
POKE SM,C0+16*C1
```

Se voleste cambiare il colore di fondo della locazione SM senza influenzare il colore di primo piano, dovreste utilizzare la seguente istruzione:

```
POKE SM,(PEEK(SM) AND 240) OR C0
```

Per cambiare il colore di primo piano senza modificare il colore di fondo usate questa istruzione:

```
POKE SM,(PEEK(SM) AND 15) OR 16*C1
```

Il formato multicolore

Esiste anche un altro formato di grafica a mappa di bit che non abbiamo ancora preso in considerazione: il formato multicolore a mappa di bit. Questo formato vi consente di aggirare la limitazione riguardante la possibilità di visualizzare solo due colori nella stessa cella. Nel formato multicolore a mappa di bit si possono visualizzare fino a quattro colori: il colore di fondo e tre colori di primo piano. Per segnalare al VIC-II di attivare il formato multicolore, dopo esserci già posti in formato grafico a mappa di bit, inserite POKE 53270, PEEK(53270) OR 16.

Come codifica i colori la mappa di bit. Dal momento che ciascun bit nella mappa di bit è attivato o disattivato, come possiamo codificare *quattro* colori? Il Commodore 64 ottiene ciò collegando

ogni due bit insieme, formando coppie di bit che operano congiuntamente. Un bit può offrire solo due scelte, attivato o disattivato. Due bit operanti insieme, invece, possono offrire quattro scelte:

00　　0
01　　1
10　　2
11　　3

Ogni coppia di bit, quindi, può specificare il colore di fondo (0) o uno dei tre colori di primo piano (1-3).

Questo significa che ci vogliono *due* bit per controllare ciascun punto. Ciò richiederebbe 16Kbyte, vale a dire l'intero banco di memoria dedicato alla grafica. Per aggirare questo problema anche i pixel sullo schermo sono accoppiati. Vale a dire, ogni coppia di bit controlla una coppia di pixel. Ciò vi consente di mantenere lo schermo multicolore negli stessi 8Kbyte della mappa di bit in formato normale.

C'è uno svantaggio. Dal momento che entrambi i pixel di una coppia sono controllati dalla stessa coppia di bit, devono avere sempre lo stesso colore. In pratica, tutti i punti dello schermo

avranno una risoluzione orizzontale pari a due pixel. Il grado di risoluzione ottenuto sarà di soli 160 per 250 punti invece di 320 per 250. Tuttavia, le ulteriori possibilità offerte dai disegni multicolori spesso ripagano pienamente della perdita nel grado di risoluzione.

AND e OR con i byte multicolori. Ogni byte nel formato multicolore consiste di quattro coppie di bit, come queste:

00 00 00 00

Per cambiare una coppia di pixel sullo schermo dovete cambiare due bit per volta, non uno.

Spesso il modo più conveniente per farlo consiste nell'organizzare una matrice colore ed una matrice delle coppie di bit. Avrete bisogno di quattro matrici colore, una per ciascun colore:

Colore 0	**00** (0)	**00**	**00**	**00**
Colore 1	**01** (85)	**01**	**01**	**01**
Colore 2	**10** (170)	**10**	**10**	**10**
Colore 3	**11** (255)	**11**	**11**	**11**

Notate che ciascuna matrice colore consiste di un byte con ogni coppia di bit inizializzata allo stesso colore. Se usaste la sola matrice colore, potreste cambiare solo interi byte per volta, non singole coppie di bit.

Avrete anche bisogno di quattro matrici a coppie di bit, una per ciascuna coppia di bit:

Coppia di bit 0	**11**	**00**	**00**	**00**
Coppia di bit 1	**00**	**11**	**00**	**00**
Coppia di bit 2	**00**	**00**	**11**	**00**
Coppia di bit 3	**00**	**00**	**00**	**11**

Notate che ciascuna matrice a coppie di bit ha solo una coppia di

bit attivata; se venissero usate direttamente, porrebbero sempre la coppia di bit in oggetto al colore 3.

Ma potete usare le matrici a coppie di bit e colore insieme, per porre la giusta coppia di bit esattamente al colore voluto, senza cambiare le altre coppie di bit del byte.

Innanzi tutto organizziamo le due serie di matrici sotto forma di vettori. Le matrici colore vanno da C(0) a C(3). Le matrici a coppie di bit vanno da BP(0) a BP(3). In questo esempio supponiamo di operare sulla coppia di bit 2; vogliamo cambiarne il colore nel colore 1. Dopo aver compiuto l'operazione, vogliamo assicurarci che l'aspetto del byte è questo:

?? ?? 01 ??

I punti di domanda rappresentano bit che non abbiamo intenzione di cambiare. Non sappiamo quale sia il loro contenuto e non ce ne preoccupiamo, a parte il fatto che vogliamo lasciarli immutati.

Nel nostro esempio, tuttavia, diremo che il byte su cui stiamo operando è stato interamente posto al colore 2. Il numero binario 10 10 10 10 ha un valore decimale di 170. Il nostro risultato dovrebbe essere il numero binario 10 10 01 10, che ha un valore decimale di 166.

Ecco come possiamo assicurarci di aver ottenuto il risultato voluto.

Innanzi tutto prendiamo il byte originale, XB, e creiamo una finestra per il nuovo colore, cancellando il contenuto attuale della coppia di bit in questione. Potete ottenere questo risultato applicando l'operatore AND all'*inverso* della matrice delle coppie di bit. L'inverso è dato da 255 *meno* la matrice delle coppie di bit. Nel nostro caso la matrice delle coppie di bit è BP(2), il numero binario 00 00 11 00, o 12 in forma decimale. Sottraiamola da 255 (valore numerico binario 11 11 11 11) ed otterremo un risultato di 243, vale a dire in binario 11 11 00 11. Quando applicate al numero l'operatore AND, con il byte XB come argomento, esso cambierà la coppia di bit 2 in zero e lascerà le altre coppie di bit indisturbate.

Ecco una riga di programma che ottiene ciò e registra la finestra nel byte risultante nella variabile WB. Nel nostro esempio WB dovrebbe essere in binario 10 10 00 10, o in decimale 162.

WB=XB AND (255-BP(2))

Ora che avete creato una finestra nel byte, per accogliere la nuova coppia di bit, dovete creare una coppia di bit del colore opportuno nella giusta posizione. Tutto ciò che bisogna fare è applicare l'operatore AND alla matrice colore ed alla matrice a coppie di bit. La matrice colore C(1) è in binario 01 01 01 01 e la matrice a coppie di bit BP(2) 00 00 11 00. Il risultato della operazione AND è il numero binario 00 00 01 00: la coppia di bit in questione è posta al colore esatto e tutti gli altri bit sono posti a 0.

Ecco una istruzione per ottenere ciò, registrando la coppia di bit finale nella variabile FP:

FP=C(1) AND BP(2)

Ora tutto quello che rimane da fare è applicare l'operatore OR alla coppia di bit finale ed al byte in cui è stata creata la finestra. Nel nostro esempio il byte contenente la finestra era 10 10 00 10 e la coppia di bit in oggetto era 00 00 01 00. Applicandovi l'operatore OR, il numero binario risultante è 10 10 01 10, il che è esattamente il risultato desiderato. In questa istruzione il risultato della operazione viene registrato nella variabile NB:

NB=FP OR WB

Tutte queste operazioni possono essere conglobate in una sola riga di programma:

WB=XB AND (255-BP(2)):FP=C(1) AND BP(2):NB=FP OR WB

O, in forma anche più semplice:

NB=(XB AND (255-BP(2))) OR (C(1) AND BP(2))

Se sono disponibili anche l'indice della coppia di bit e l'indice del colore (BN e CN), la riga seguente traccerebbe la giusta coppia di bit in qualsiasi programma multicolore e a mappa di bit:

NB=(XB AND (255-BP(BN))) OR (C(CN) AND BP(BN))

In effetti, se la variabile MM è inizializzata all'indirizzo assoluto

del byte che desiderate cambiare, potete anche eliminare sia NB che XB:

POKE MM,(PEEK(MM)) AND (255-BP(BN)) OR (C(CN) AND BP(BN))

Questa riga verrà eseguita tanto rapidamente quanto è nelle vostre speranze.

Come si cambiano i colori in formato multicolore. Oltre a dover maneggiare le coppie di bit, esiste un altro problema con il formato multicolore. La memoria di schermo può contenere solo due codici colore per byte, uno nei quattro bit di sinistra ed il colore di fondo nei quattro bit di destra. Dove assegnare gli altri due colori disponibili in una cella di mappa di bit multicolore?

Dal momento che utilizziamo la memoria di schermo per gli assegnamenti dei colori nel formato grafico a mappa di bit, abbiamo ancora a disposizione la normale memoria colore, che parte dalla locazione 55296. La memoria colore è organizzata nello stesso ordine della memoria di schermo e delle celle internamente alla mappa di bit.

Tuttavia, solo i quattro bit meno significativi (bit 0-3) hanno una funzione effettiva nella memoria colore, quindi abbiamo un ulteriore colore da assegnare. A questo scopo utilizziamo il registro colore di fondo del chip VIC-II in 53281. Sfortunatamente, questo significa che tutte le celle hanno lo stesso colore di fondo, a meno che utilizziate interruzioni di scansione video (vedi «Come utilizzare formati grafici diversi sul C64»). Tuttavia, gli altri tre colori possono essere individualmente assegnati per ciascuna cella, il che vi dà diverse possibilità per realizzare combinazioni di colori.

Il colore richiamato dalla coppia di bit 00 (colore 0) sarà il colore di fondo, che è contenuto nel registro colore di fondo in 53281.

Il colore richiamato dalla coppia di bit 01 (colore 1) sarà il colore contenuto nei quattro bit di sinistra (bit 4-7) del byte corrispondente nella memoria di schermo.

Il colore richiamato dalla coppia di bit 10 (colore 2) sarà il colore contenuto nei quattro bit di destra (bit 0-3) del corrispondente byte nella memoria di schermo.

Il colore richiamato dalla coppia di bit 11 (colore 3) sarà il colore contenuto nei quattro bit di destra (bit 0-3) del byte corrispondente nella memoria di schermo.

Questo può risultare particolarmente sconcertante, se desideraste cambiare i colori a metà programma, a parte il fatto che le celle della mappa di bit, i byte della memoria di schermo e quelli della memoria colore sono disposti esattamente nello stesso ordine. Questo significa che gli stessi valori numerici esprimenti gli scarti possono essere utilizzati per trovare i byte della memoria di schermo ed i byte della memoria colore che controllano i colori di una particolare cella. Supponiamo che vogliate cambiare i colori attribuiti al byte 683 della mappa di bit. Lo scarto della cella è INT(683/8), cioè 85. Quindi, per cambiare il colore attribuito al byte numero 683 dall'inizio della mappa di bit dovreste cambiare il byte 85 dall'inizio della memoria colore e/o il byte 85 dall'inizio della memoria di schermo.

Come far ritornare il vostro C64 alla normalità

Per ripristinare il vostro calcolatore alle normali operazioni usate i seguenti comandi:

POKE 53265,27:POKE 53270,200:POKE 53272,20:POKE 56576,151

Come proteggere i disegni da voi realizzati

Quando si usa il BASIC ed il formato a mappa di bit, il BASIC può avere una certa tendenza a traboccare e ad usare la memoria a mappa di bit, o la memoria di schermo, per registrare righe di programma o variabili. Per impedire ciò dovete ingannare il BASIC, in modo che pensi che la memoria del calcolatore termini *prima* di raggiungere la vostra mappa. Ecco perché non dovreste usare il banco grafico 0 come schermo grafico a mappa di bit; non ci sarebbe quasi posto per un qualsiasi tipo di programma, se il BASIC dovesse dividere il blocco di 16Kbyte con la mappa di bit.

Per cambiare il punto in cui il BASIC pensa che la memoria termini dovete inserire nuovi valori numerici nelle locazioni 55 e 56, tramite istruzioni POKE. L'estremità superiore della memoria disponibile dovrebbe essere predisposta all'indirizzo *più basso* da voi usato nel blocco riservato alla grafica. Se fate partire la mappa di bit, diciamo, dalla locazione 16384, questo numero sarebbe quindi quello utilizzato come estremità superiore della memoria. Se fate

partire la memoria della mappa di bit nel blocco a mappa di bit 1 e la memoria di schermo al blocco di memoria di schermo 7, potete lasciare che il BASIC usi la memoria fino all'indirizzo 23551; il nuovo indirizzo del termine della memoria sarà 23552, che rappresenta il primo indirizzo nel blocco della memoria di schermo.

Il numero 23552 è troppo grande per poterlo inserire in una qualsiasi locazione di memoria, dal momento che nessuna locazione può contenere più di un byte da otto bit. Il numero più grande che una qualsiasi locazione può contenere è 255. Tutti gli indirizzi di un calcolatore, tuttavia, tranne i primi 255, sono contraddistinti da numeri superiori a questo. Il computer li tratta spezzando l'indirizzo in due byte. La parte meno significativa dell'indirizzo che state registrando è quasi sempre piazzata prima della parte più significativa.

L'indirizzo 23552 è un numero binario a 16 bit 0101110000000000. Questo numero è suddiviso in due metà, 01011100 (decimale 92) e 00000000 (decimale 0). La parte meno significativa dell'indirizzo, 0, viene inserita nella locazione 55; la parte più significativa, 92, nella locazione 56.

Non è necessario, tuttavia, che calcoliate i numeri binari. Invece questa riga di programma suddividerà qualsiasi numero intero XX compreso fra 0 e 65535 nel byte meno significativo (LB) e nel byte più significativo (HB):

```
HB=INT(XX/256):LB=XX-HB*256
```

Quindi il vostro programma richiede solo di inserire, tramite una istruzione POKE, LB nella prima locazione di memoria e HB nella seconda locazione di memoria. È probabile che siate già pratici di questa tecnica; ogni calcolatore che impieghi le CPU 6502 o 6510 la utilizza di frequente.

Effetti speciali

Quando create disegni usando una mappa di bit, gran parte del loro aspetto dipende da ciò che ne fate.

Potete inizializzare i puntatori alla vostra mappa di bit ed alla memoria di schermo *prima* che il vostro programma tracci il disegno. In questo modo l'utente può vedere il disegno mentre viene tracciato.

Potete tracciare il disegno nell'area della mappa di bit, mentre i puntatori continuano ad indicare il set grafico standard. Quindi, quando cambiate i puntatori l'intero disegno appare improvvisamente sullo schermo. Dà l'effetto della velocità del lampo, persino in BASIC.

Potete preparare due o tre mappe di bit ed alternarle cambiando i puntatori. Questo richiede parecchia memoria, ma l'effetto può essere splendido e l'attivazione dei vari disegni è quasi istantanea in BASIC.

Poiché ogni cella ha esattamente le stesse dimensioni dei caratteri del set contenuto in memoria, potete facilmente porre lettere e caratteri sullo schermo ad alta risoluzione, estraendo i caratteri dal set di memoria con una PEEK ed inserendoli nelle singole celle, nello stesso ordine, con una POKE.

Potete supplire ai colori dello schermo a mappa di bit usando gli sprite, che possono avere fino a tre colori ciascuno. Tra l'altro non è necessario che gli sprite si muovano. Combinando sprite e grafica ad alta risoluzione potete ottenere disegni molto realistici, ricchi di dettagli.

I codici colore nella memoria di schermo (e nella memoria colore, per il formato multicolore) consistono ciascuno di un byte, mentre le celle della mappa di bit sono di otto byte. Quindi potete cambiare la memoria colore e di schermo molto più velocemente di quanto cambiate mappa di bit. Se cercate di realizzare delle animazioni in BASIC, linguaggio in cui la velocità rappresenta un costante problema, potete ottenere una maggiore rapidità cambiando i *colori* piuttosto che spostando i *pixel*. Ciò non sarà possibile per la maggior parte delle animazioni, naturalmente, ma nei casi in cui funziona l'aumento di velocità può essere notevole.

Un sottoprogramma che disegna in linguaggio macchina

Ecco un breve sottoprogramma in linguaggio macchina che eseguirà quattro comandi sullo schermo a mappa di bit bicolore. Potete aggiungerlo ai vostri programmi BASIC e mandarlo in esecuzione tramite l'istruzione SYS AD, dove AD è l'indirizzo in cui il vostro programma ha inserito in memoria, con una POKE, il primo byte del sottoprogramma in linguaggio macchina.

Il comando SYS deve essere seguito da un numero, da 0 a 3, che precisi il comando desiderato. In aggiunta, alcuni dei comandi

richiedono di includere altri numeri nel comando SYS. Viene fornito un esempio per ciascun comando:

Comando 0: cancella lo schermo. Formato:
SYS AD, 0

Questo comando cancella lo schermo a mappa di bit, ponendo tutti i byte della mappa di bit a 0.

Comando 1: assegna i colori. Formato:
SYS AD,1, *nn*

Questo comando pone tutti i byte della memoria di schermo al valore *nn*. Ciò vi consente di assegnare immediatamente tutti i colori di ogni cella della mappa di bit alla velocità tipica del linguaggio macchina. Ricordatevi che i bit da 4 a 7 del numero *nn* controllano i colori di primo piano ed i bit da 0 a 3 il colore di fondo.

Comando 2: traccia un punto. Formato:
SYS AD,2, *xx,yy*

Questo comando pone un singolo punto sullo schermo alla locazione contraddistinta dai valori *xx* ed *yy*. Il valore numerico xx rappresenta la colonna (posizione orizzontale) del pixel in oggetto e deve essere un numero da 0 a 319. Il numero *yy* rappresenta l'indice di riga (posizione verticale) del pixel voluto e deve essere un valore numerico compreso tra 0 e 249.

Comando 3: cancella un punto. Formato:
SYS AD,3, *xx,yy*

Questo comando è identico al comando 2, a parte il fatto che invece di porre il pixel a 1 lo annulla.

Prima di poter usare il sottoprogramma il vostro programma BASIC deve comunicare al calcolatore dove iniziano la memoria di schermo e la mappa di bit. Inserite l'indirizzo di partenza della mappa, diviso per 256, nella locazione 680 con una POKE. Inserite l'indirizzo di partenza della memoria di schermo, diviso 256, nella locazione 681. Non dovete inserire in memoria i byte meno significativi di questi indirizzi, poiché il sottoprogramma «sa» che i byte

meno significativi saranno sempre uguali a zero.

Le righe 10-30 leggono le istruzioni DATA ed inseriscono in memoria il sottoprogramma in linguaggio macchina. Potete inserire il sottoprogramma in un altro punto, ma è bene porlo in un'area di memoria protetta.

Le righe 100-300 rappresentano il sottoprogramma in linguaggio macchina sotto forma di istruzioni DATA. Dovete fare molta attenzione nel copiarle. È facile commettere errori di battitura quando si copiano righe su righe di numeri. Se il sottoprogramma non funziona, controllate innanzi tutto le istruzioni DATA per accertare eventuali errori.

Le righe 500-600 rappresentano un programma esemplificativo, che utilizza il sottoprogramma in linguaggio macchina per tracciare una curva sinusoidale. Non includete queste righe nei vostri programmi.

Bitmap utility

```
1 REM *PROGRAMMA DI UTILITA' A MAPPA DI BI
  T*
3 REM COMANDI:
4 REM XX SYS (BASE),OPZIONE,DATI
5 REM OPZIONI:
6 REM SYS B, 0 - CANCELLA LO SCHERMO
7 REM SYS B, 1, CL - ASSEGNA IL COLORE DEF
  INITO DAL VALORE NUMERICO CL
8 REM SYS B, 2, X, Y - TRACCIA IL PUNTO (X
  ,Y)
9 REM SYS B, 3, X, Y - CANCELLA IL PUNTO (
  X,Y)
10 AD=32768:REM ** INDIRIZZO DI BASE **
20 READD:IFD=-1THEN500:REM ** RICHIAMA LA
   ROUTINE D'UTENTE
30 POKEAD,D:AD=AD+1:GOTO20
100 DATA32,115,0,32,158,173,32,247,183,140
    ,170,2,192,0
110 DATA240,6,192,1,240,32,208,77,173,168,
    2,133,252,24
```

```
120 DATA105,32,133,253,169,0,133,251,168,1
    45,251,230,251,208
130 DATA2,230,252,166,252,228,253,144,242,
    96,32,115,0,32
140 DATA158,173,32,247,183,132,253,173,169
    ,2,56,233,1,133
150 DATA252,24,105,4,133,254,169,8,133,251
    ,160,247,165,253
160 DATA145,251,230,251,208,2,230,252,166,
    252,228,254,144,242
170 DATA96,32,115,0,32,158,173,32,247,183,
    140,171,2,141
180 DATA172,2,32,115,0,32,158,173,32,247,1
    83,140,173,2
190 DATA152,41,248,133,253,141,180,2,141,1
    74,2,169,0,133
200 DATA254,141,181,2,162,4,24,38,253,38,2
    54,202,16,248
210 DATA162,2,24,46,180,2,46,181,2,202,16,
    246,24,165
220 DATA253,109,180,2,141,178,2,165,254,10
    9,181,2,141,179
230 DATA2,173,171,2,41,248,141,176,2,173,1
    72,2,141,177
240 DATA2,56,173,173,2,237,174,2,24,109,17
    6,2,133,251
250 DATA173,177,2,109,168,2,133,252,24,173
    ,178,2,101,251
260 DATA133,251,173,179,2,101,252,133,252,
    56,173,171,2,237
270 DATA176,2,133,253,56,162,255,169,0,106
    ,232,228,253,208
280 DATA250,141,180,2,174,170,2,224,3,240,
    10,160,0,177
290 DATA251,13,180,2,145,251,96,56,169,255
    ,237,180,2,141
300 DATA180,2,160,0,177,251,45,180,2,145,2
    51,96,-1
500 REM ** ROUTINE D'UTENTE **
```

```
501 REM DISEGNA UNA CURVA SINUSOIDALE
505 POKE53265,PEEK(53265)OR2↑5:REM ** ATTI
    VA IL METODO BIT MAP
510 POKE680,96:POKE681,92:REM ** INIZIALIZ
    ZA I PUNTATORI ALLA ROUTINE
515 POKE53272,120:POKE56576,2:REM ** ATTIV
    A LA MEMORIA DEL CHIP VIC-II
520 POKE55,0:POKE56,60:CLR:REM ** PROTEGGE
     LA MAPPA DI BIT DAL PROGRAMMA BASIC
530 B=32768:REM ** INIZIALIZZA L'INDIRIZZO
     DI BASE DELLA ROUTINE
540 SYS B,0:SYS B,1,16:REM ** CANCELLA LO
    SCHERMO E ASSEGNA IL COLORE
550 FORX=0TO6.4STEP.02:Y=SIN(X):REM ** CAL
    COLA IL VALORE DELLA CURVA SINUSOIDALE

560 X1=X*50:Y=Y*50:REM ** AMPLIFICA LE DIM
    ENSIONI DEL GRAFICO
570 Y=100-Y:SYS B,2,X1,Y:REM   ** TRACCIA
    UN PUNTO
580 NEXT:REM ** PUNTO SUCCESSIVO
590 GOTO590
600 REM ** SI ESCE PREMENDO RUN-STOP/RESTO
    RE
```

Capitolo due

Arte istantanea

Bob Urso

Entrambi questi programmi grafici per il Commodore 64 — uno casuale, l'altro controllato dall'utente — creano disegni espressivi ed eleganti.

Chiunque veda il vostro C64 mentre state eseguendo uno di questi due programmi potrebbe pensare che abbiate appena saccheggiato il Museo d'Arte Moderna. Ciascun programma vi permette di creare grafici espressivi e coloratissimi.

Il programma 1 realizza grafici in modo completamente casuale. La scelta del colore, della direzione e dei caratteri viene effettuata nelle righe 30-89. L'inserimento dei caratteri tramite una istruzione POKE e l'aggiornamento della posizione per il prossimo ciclo vengono trattati dalla riga 90. Le righe 95 e 96 mantengono il disegno all'interno dei limiti di schermo.

Il tempo (riga 11) viene inizializzato a 1000; una volta che questo numero è stato raggiunto lo schermo viene cancellato, dopo essere stato parzialmente riempito dal disegno casuale. Potete incrementare T, per far si che il vostro disegno possa ulteriormente complicarsi, oppure potete eliminare le righe 11 e 99-120 ed il disegno continuerà a riempire lo schermo fino al successivo spegnimento.

Il secondo programma si chiama «Schizzo»; *vi* permette di disegnare effettivamente. Potete cambiare i colori premendo i tasti colore, senza dover premere CTRL. I tasti di selezione dei caratteri sono raggruppati sulla sinistra, in modo che non interferiscano con i tasti di selezione delle direzioni.

Potete muovervi in otto direzioni, che permettono di tracciare linee sia diagonali che orizzontali e verticali. Una volta premuto un tasto direzionale, il disegno continuerà in quella direzione finché non raggiungerà il bordo dello schermo, oppure fino a che premerete un qualsiasi altro tasto per interromperlo.

Ci sono dubbi che riusciate ad ingelosire un Rembrandt, ma sarete più che ricompensati per il poco tempo richiesto per copiare questo programma.

Capitolo due

Programma 1. Sottoprogramma di grafica casuale

```
10 REM GRAFICA RANDOM
11 T=1000
15 PRINT"{CLR}"
17 POKE53280,0:POKE53281,0
20 P=1024+INT(RND(1)*999)+1:G=P+54272
30 Z=INT(5*RND(1))+1
40 IFZ=1THENS=81
41 IFZ=2THENS=64
42 IFZ=3THENS=84
43 IFZ=4THENS=102
44 IFZ=5THENS=160
45 K=INT(8*RND(1))+1
50 IFK=1THENC=9
51 IFK=2THENC=1
52 IFK=3THENC=2
53 IFK=4THENC=3
54 IFK=5THENC=4
55 IFK=6THENC=5
56 IFK=7THENC=6
57 IFK=8THENC=7
80 D=INT(8*RND(1))+1
81 IFD=1THENR=-39
82 IFD=2THENR=-40
83 IFD=3THENR=-41
84 IFD=4THENR=-1
85 IFD=5THENR=1
86 IFD=6THENR=39
87 IFD=7THENR=40
88 IFD=8THENR=41
89 M=INT(40*RND(1))+1
90 FORZ=1TOM:POKEP,S:POKEG,C:P=P+R
95 IFP<=1024THENP=P-R
96 IFP>=2023 THEN P=P-R
97 G=P+54272
99 T=T-1
100 IFT=0THENGOTO10
```

```
110 PRINT"TEMPO";T
120 PRINT"{ 3 SU}"
1101 NEXTZ
1110 GOTO30
```

Programma 2. Schizzo

```
10 REM SCHIZZO - DI BOB URSO
20 P=1524:S=160:C=1
90 POKE53280,0:POKE53281,0
95 GOTO1000
99 PRINT"{CLR}"
100 G=P+54272
200 POKE P,S :POKEG,C
300 GET G$:IFA$<>G$ANDG$<>""THENA$=G$
310 IFA$="I"THENP=P-40
320 IFA$="U"THENP=P-41
330 IFA$="O"THENP=P-39
340 IFA$="J"THENP=P-1
350 IFA$="K"THENP=P+1
360 IFA$="N"THENP=P+39
365 IFA$="M"THENP=P+40
370 IFA$=","THENP=P+41
380 IFA$="1"THENC=0
390 IFA$="2"THENC=1
400 IFA$="3"THENC=2
410 IFA$="4"THENC=3
420 IFA$="5"THENC=4
430 IFA$="6"THENC=5
440 IFA$="7"THENC=6
450 IFA$="8"THENC=7
460 IFA$="Q"THENS=81
470 IFA$="A"THENS=64
480 IFA$="Z"THENS=66
490 IFA$="W"THENS=102
500 IFA$="S"THENS=160
510 FORZ=1024TO1984STEP40:IFP=ZTHENP=P+1
530 IFP<1024THENP=P+40
540 IFP>2023THENP=P-40
550 GOTO 100
```

```
1000 PRINT"{CLR}":PRINT"{ 2 GIU'}
     { 5 SPAZI}SCHIZZO{ 4 SPAZI}DI BOB URS
     O{ 4 SPAZI}10/82":PRINT"{GIU'}"
1010 PRINT"QUESTI SONO I CARATTERI CHE PUO
     I USARE"
1020 PRINT"{ 3 SPAZI}PREMI Q PER Q"
1021 PRINT"{ 3 SPAZI}PREMI A PER C"
1022 PRINT"{ 3 SPAZI}PREMI Z PER B"
1023 PRINT"{ 3 SPAZI}PREMI W PER [<X>]"
1024 PRINT"{ 3 SPAZI}PREMI S PER {RVS}
     {OFF}"
1030 PRINT"{GRN}PER CAMBIARE I COLORI INDI
     CATI SUI TASTI"
1040 PRINT"PREMI IL CORRISPONDENTE NUMERO
     DA 1 A 8":PRINT"{GIU'}"
1070 PRINT"[<7>]PER MUOVERE IL CARATTERE U
     SA"
1080 PRINT"{ 10 SPAZI}U{ 2 SPAZI}I
     { 2 SPAZI}O"
1090 PRINT"{ 11 SPAZI}M ↑ N"
1100 PRINT"{ 10 SPAZI}J← Q *K"
1110 PRINT"{ 11 SPAZI}N B M"
1120 PRINT"{ 10 SPAZI}N{ 2 SPAZI}M
     { 2 SPAZI},"
1130 PRINT"{PUR}PER FERMARLO, PREMI UN TAS
     TO COLORE"
1150 PRINT"[<7>]ISTRUZIONI TERMINATE ?PREM
     I S"
1160 GETR$:IF R$="" AND R$<>"S"THEN1160
1170 IFR$="S"GOTO99
```

Capitolo due

Il formato «Colore di fondo esteso»

Sheldon Leemon

Il formato colore di fondo esteso può essere uno strumento molto utile quando desiderate creare delle schermate multicolori. Questo articolo, oltre ad illustrare come si usa questo formato, comprende anche un breve programma che vi aiuta a scegliere le più opportune combinazioni di colori.

È noto come sia possibile scegliere singolarmente i colori di primo piano di ogni lettera sullo schermo di testo del Commodore 64. Meno noto è il fatto che potete scegliere singolarmente anche i colori di fondo. Ciò è reso possibile dal formato colore di fondo esteso del C64. Benché questo formato di visualizzazione non venga affatto citato nel *Manuale d'uso,* e venga trattato molto brevemente nella *Guida di riferimento del programmatore,* vale la pena di esaminarlo più approfonditamente. Vediamo come viene usato e quali sono le differenze rispetto al formato di visualizzazione di testo normale.

Normalmente ci sono 256 forme di caratteri che possono essere visualizzate sullo schermo. Potete vederli sia usando l'istruzione PRINT sia inserendo un codice di visualizzazione da 0 a 255 nella memoria di schermo, tramite una istruzione POKE, ed un codice colore da 0 a 15 nella memoria colore (ad esempio una coppia di istruzioni POKE 1024,1 e POKE 55296,1 farà apparire una lettera A bianca nell'angolo superiore sinistro dello schermo). Il colore di fondo dello schermo è stabilito dal Registro colore di fondo 0, in 53281. Potete cambiare questo colore di fondo inserendo, con una POKE, un nuovo valore in 53281. Ad esempio, POKE 53281,0 crea un fondo nero.

Quando viene attivato il formato colore di fondo esteso, tuttavia, il numero di forme di caratteri che possono essere visualizzate si riduce a 64; sono le prime 64 forme reperibili nella tavola dei simboli di visualizzazione dello schermo (Appendice G). Questo gruppo comprende le lettere dell'alfabeto, i numeri ed i segni di

interpunzione.

Se cercate di visualizzare sullo schermo un carattere con un codice simbolico più grande, la forma che verrà mostrata apparterrà al primo gruppo di 64 caratteri, ma il colore di fondo del carattere non sarà più stabilito dal registro della locazione 53281; verrà invece fissato da uno degli altri registri colore di fondo. I caratteri aventi codici simbolici compresi fra 64 e 127 ricaveranno il proprio colore di fondo dal registro 1, contenuto nella locazione 53282. Questi caratteri comprendono i caratteri alfanumerici maiuscoli ed altri caratteri grafici. Quelli aventi codice da 128 a 191 avranno colore di fondo determinato dal registro 2, in 53283. Questi comprendono cifre, lettere e segni di interpunzione in negativo. Infine i codici carattere da 192 a 255 useranno il registro 3, in 53284. Questi sono rappresentati dai caratteri grafici in negativo.

Proviamo a sperimentarne il funzionamento. Innanzi tutto visualizzeremo quattro lettere sullo schermo:

```
FOR I=0 TO 3:POKE 1230+(I*8),I*64
+1:POKE 55502+(I*8)1:NEXT
```

Dovrebbero apparire sullo schermo quattro lettere bianche, una a, una A maiuscola, una a in negativo, una A maiuscola in negativo, tutte su fondo blu. Quindi inseriremo i colori negli altri registri colore di fondo:

```
POKE 53282,0:POKE 53283,2:POKE 53284,5
```

Questi comandi inizializzano i registri suddetti a nero, rosso e verde, rispettivamente. Infine attiveremo il formato colore di fondo esteso. Si ottiene ciò ponendo a 1 il bit 6 del registro VIC-II nella locazione 53265. Quindi, per attivare questo formato usiamo l'istruzione:

```
POKE 53265,PEEK(53265) OR 64
```

Noterete che sono accadute due cose. La prima: tutte le lettere assumono lo stesso aspetto, quello di una A maiuscola. La seconda: ciascuna lettera prende il colore di fondo di un differente registro colore. Per far ritornare le cose alla normalità disattivate il formato colore di fondo esteso con questa istruzione:

```
POKE 53265,PEEK(53265) AND 191
```

Il formato colore di fondo esteso può rappresentare un utilissimo arricchimento delle vostre schermate di testo. Consente la creazione di finestre, che, a causa dei differenti colori di fondo, consentono a differenti corpi di testo di stagliarsi visivamente in maniera distinta. Ad esempio, il testo di un programma di avventura potrebbe avere una finestra per mostrare la posizione attuale del giocatore, una per mostrare l'inventario degli oggetti posseduti ed una per acquisire i comandi per le mosse successive. Una finestra può essere fatta lampeggiare, in certe occasioni, per richiamare l'attenzione su di un particolare messaggio, inserendo semplicemente un nuovo valore nei registri colore tramite una POKE. E facendo variare i colori di primo piano si può far scomparire sia la finestra che il messaggio che contiene e farli riapparire in seguito.

Come superare le difficoltà

Esistono, tuttavia, un paio di problemi relativi all'uso di queste finestre. La forma dei caratteri che volete usare può non avere un codice simbolico inferiore a 64. In questo caso la soluzione consiste nel definire un proprio set di caratteri, nel quale le forme di caratteri volute sono comprese nel primo gruppo di 64.

Un altro problema è rappresentato dal fatto che i caratteri contenuti in una istruzione PRINT del vostro programma non sempre avranno lo stesso aspetto sullo schermo. Essere costretti a calcolare quale lettera battere per ottenere il carattere 4 può essere piuttosto scomodo. La soluzione più semplice di questo problema può consistere in un sottoprogramma che operi per voi la traduzione. Dal momento che le lettere appaiono in formato normale nella finestra 1, mentre i caratteri della finestra 3 sono semplicemente i caratteri della finestra 1 in negativo, avrete problemi solo con i caratteri nelle finestre 2 e 4. Per convertire questi caratteri inserite i vostri messaggi nella variabile alfanumerica A$, ed usate il seguente sottoprogramma:

```
500 B$="":FOR I=1 TO LEN(A$):
    B=ASC(MID$(A$,I,1))
510 B=B+32:IF B<96 THEN B=B+96
520 B$=B$+CHR$(B):NEXT I:RETURN
```

Questo programma converte ciascuna lettera nel suo equivalente codice ASCII, aggiunge lo scarto opportuno e lo riconverte in parte di una nuova variabile alfanumerica, B$. Quando la conversione è stata completata, B$ conterrà i caratteri necessari per stampare il messaggio con una istruzione PRINT, nella finestra 2. Per quanto riguarda la finestra 4, PRINT CHR$(18); B$; CHR$(146). Questa istruzione attiverà il negativo prima di visualizzare la variabile alfanumerica e lo disattiverà in seguito.

Un'altra cosa a cui dovete badare è la posizione del cursore prima dell'uso di una istruzione PRINT, per assicurarvi di stampare all'interno della finestra. La posizione orizzontale è facilmente controllabile; potete usare l'istruzione TAB per spostare il cursore sulla colonna opportuna. La posizione verticale è di controllo un po' più difficile, dal momento che non esiste nessun comando specifico per governarla. Una soluzione consiste nel richiamare il cursore nella posizione di base (HOME) ed utilizzare un numero opportuno di comandi CRSR verso il basso. Un modo particolarmente pratico per fare ciò consiste nel creare un vettore alfanumerico, con ciascuna variabile contenente il carattere HOME ed un numero sufficiente di caratteri CRSR verso il basso a posizionarsi sulla riga voluta. L'istruzione:

```
DIM RO$(25): RO$(0)=CHR$(19):FOR I=1
TO 24:RO$(I)=RO$(I-1)+CHR$(17):
NEXT
```

fornisce un simile vettore. Se volete visualizzare un messaggio sulla riga 10, potete limitarvi all'istruzione PRINT RO$(10); «HELLO».

Alcuni esempi pratici

Una dimostrazione pratica della tecnica necessaria per predisporre le finestre è fornita dal programma 1. Il programma prepara tre finestre e le mostra lampeggiare, le fa apparire e scomparire.

Il programma 2 risolve un altro problema reale: che colori scegliere come primo piano e come fondo per creare un opportuno contrasto, al fine di ottenere una buona leggibilità. Ciò rappresenta un problema molto più sentito sul C64 che sul VIC, dove le lettere sono molto più grandi. Quindi, con l'aiuto di un po' di magia in linguaggio macchina il programma 2 organizza sullo schermo una tavola che mostra quali sono le migliori combinazioni. Mostra

simultaneamente tutte le 256 combinazioni di colori di fondo e di primo piano. I colori di fondo vanno da 0 sulla riga superiore a 15 sull'ultima riga ed i colori di primo piano da 0 a sinistra a 15 a destra. Si ottiene ciò tramite il registro di scansione video, che segnala al calcolatore qual è la riga attualmente scandita, in modo che sappia quando cambiare il colore di fondo a metà schermo. Dal momento che il programma è in ciclo chiuso, l'unico modo di uscirne consiste nel premere contemporaneamente i tasti STOP e RESTORE. Registrate una copia di questo programma prima di mandarlo in esecuzione.

Programma 1. Finestre

```
1 REM **** FINESTRE ****
5 DIMRO$(25):RO$(0)=CHR$(19):FORI=1TO24:RO
  $(I)=RO$(I-1)+CHR$(17):NEXT
10 POKE53265,PEEK(53265)OR64
20 POKE53280,0:POKE53281,0:POKE53282,1:POK
   E53283,2:POKE53284,13
25 OP$=CHR$(160):FORI=1TO4:OP$=OP$+OP$:NEX
   TI:PRINTCHR$(147);RO$(3);
30 FORI=1TO10:PRINTTAB(1);CHR$(18);"
   { 15 SPAZI}";TAB(23)OP$:NEXT
40 PRINTCHR$(146):PRINT:PRINT:FORI=1TO4:PR
   INTOP$;OP$;OP$;OP$;OP$;:NEXTI
50 PRINTRO$(5);CHR$(5);CHR$(18);TAB(1);"SI
    POTREBBE U-"
60 PRINTCHR$(18);TAB(1);"SARE UNA FINE-"
70 PRINTCHR$(18);TAB(1);"STRA ROSSA PER"
80 PRINTCHR$(18);TAB(1);"I MESSAGGI DI"
90 PRINTCHR$(18);TAB(1);"ERRORE."
100 A$="SI POTREBBE USA-":GOSUB300:PRINTRO
    $(5);CHR$(144);CHR$(18);TAB(23);B$
110 A$="RE UNA FINESTRA":GOSUB300:PRINTTAB
    (23);CHR$(18);B$
120 A$="VERDE PER DARE":GOSUB300:PRINTTAB(
    23);CHR$(18);B$
130 A$="ISTRUZIONI.":GOSUB300:PRINTTAB(23)
    ;CHR$(18);B$
```

```
140 PRINTCHR$(31);RO$(19);
150 A$=" MENTRE LA FINESTRA PRINCIPALE POT
    REBBE":GOSUB300:PRINTB$
160 A$=" ESSERE USATA PER ACCETTARE I COMA
    NDI.":GOSUB300:PRINTB$
170 FORI=1TO5000:NEXTI:POKE53248,0
180 FORI=1TO5:FORJ=1TO300:NEXTJ:POKE53282,
    15
190 FORJ=1TO300:NEXTJ:POKE53282,1
200 NEXTI:POKE53283,-2*(PEEK(53283)=240):P
    OKE53284,-13*(PEEK(53284)=240)
210 GOTO180
300 B$="":FORI=1TOLEN(A$):B=ASC(MID$(A$,I,
    1))
310 B=B+32:IFB<96THENB=B+96
320 B$=B$+CHR$(B):NEXTI:RETURN
```

Programma 2. Mappa colori

```
20 REM *** TAVOLA COLORI ***
30 REM
40 FORI=49152TO49188:READA:POKEI,A:NEXT
50 PRINTCHR$(147):FORI=1024TOI+1000:POKEI,
   160:POKEI+54272,11:NEXTI
60 FORI=0TO15:FORJ=0TO15
70 P=1196+(40*I)+J:POKEP,J+1:POKEP+54272,J
   :NEXTJ,I
80 SYS12*4096
100 DATA169,90,133,251,169,0,141,33,208,16
    2,15,120,173,17,208,48,251,173,18,208
110 DATA197,251,208,249,238,33,208,24,105,
    8,133,251,202,16,233,48,219
```

Capitolo due

Come utilizzare formati grafici diversi sul C64

Sheldon Leemon

Sul video del 64 è possibile utilizzare simultaneamente parecchi formati grafici differenti. Il primo programma mostra come suddividere lo schermo in tre zone: alta risoluzione, testo normale e formato grafico «bitmap» (in cui ogni bit di memoria corrisponde ad un punto — o «pixel» — dello schermo) a più colori.

Il programma 2 usa gli stessi programmi di utilità pratica, ma crea effetti completamente differenti. Lo schermo presenta tutti e tre i formati di testo: normale, a colore di fondo esteso e multicolore.

Questa tecnica grafica vi permette ampie possibilità di controllo su ciò che compare sul vostro schermo. Ad esempio, potete passare da un formato grafico all'altro con delle semplici istruzioni POKE. Pur avendo questo articolo abbondanza di riferimenti tecnici per programmatori esperti, l'autore ha fornito istruzioni e programmi esplicativi che i principianti sono in grado di seguire. Chiunque può trarre frutto da queste importanti tecniche di programmazione.

La *Guida di riferimento del Programmatore per il Commodore 64* suggerisce che è possibile far apparire sullo schermo più di un formato grafico contemporaneamente. Al momento di spiegare come si può ottenere ciò, tuttavia, il *manuale* si limita ad affermare che è necessario provocare una interruzione di cursore schermo per la riga in cui volete dare il via ad un diverso tipo di grafica (inizializzando il chip VIC-II per il nuovo formato grafico durante questa interruzione), e quindi organizzare un'altra interruzione per cambiare nuovamente il tipo di grafica un po' più avanti sul video. Questa spiegazione può essere chiara a programmatori esperti di

linguaggio macchina, ma lascia all'oscuro tutti gli altri.

In questo programma educativo tratteremo alcuni esempi di interruzione di cursore schermo che possono essere facilmente usati da programmatori BASIC per creare suddivisioni dello schermo ed altri effetti. Discuteremo anche, più dettagliatamente, su come i programmatori di linguaggio macchina possono sfruttare le potenzialità delle interruzioni di cursore schermo.

L'interruzione

Il punto più ovvio da cui partire per la nostra discussione è la spiegazione di che cosa sia una interruzione. Una interruzione (interrupt in inglese) è un segnale inviato al microprocessore (il «cervello» del calcolatore) che gli ordina di interrompere l'esecuzione del suo programma in linguaggio macchina (ad esempio, l'interprete BASIC stesso è un programma in linguaggio macchina) e di lavorare su di un altro programma per breve tempo, a volte solo una frazione di secondo. Dopo aver svolto il programma di interruzione il calcolatore ritorna al programma principale, come se non ci fossero state deviazioni nel suo corso.

C'è più di un modo di provocare un'interruzione di questo tipo sul 64. Premendo il tasto RESTORE si provoca una interruzione; se viene premuto contemporaneamente il tasto STOP, il sottoprogramma di interruzione annulla lo schermo e lo riporta al suo stato di partenza. Ci sono dei temporizzatori interni al chip CIA di ingresso/uscita, ognuno dei quali può generare delle interruzioni. Uno di questi temporizzatori è fissato dal sistema operativo in modo che provochi una interruzione ogni sessantesimo di secondo, e il sottoprogramma di interruzione che viene richiamato è utilizzato per controllare la tastiera e per aggiornare gli orologi a frequenza di un jiffy (un sessantesimo di secondo appunto) utilizzati dalle variabili interne TI e TI$. In aggiunta a ciò il chip denominato VIC-II può anche interrompere la normale esecuzione di un programma quando si verifica una di un certo numero di combinazioni correlate con l'aspetto grafico. Una di queste è chiamata interruzione di cursore schermo.

Su un normale video TV un raggio di elettroni (cursore schermo, raster, in inglese) scandisce lo schermo, partendo dall'angolo superiore sinistro e muovendosi in linea retta verso destra, illuminando parti opportune della riga di schermo durante il percorso, Quando

raggiunge il margine destro il raggio scende di una riga e ricomincia da sinistra. Ci sono 263 righe di questo genere che vengono scandite dal video del C64, 200 delle quali formano l'area video visibile. Questo raggio aggiorna lo schermo video 60 volte al secondo. Il chip VIC-II possiede registri di memoria che conservano traccia della riga che il raggio sta scandendo in un dato momento. Poiché il numero di riga può essere superiore a 255, un solo registro non è sufficiente per assolvere questo compito. Di conseguenza la parte del numero meno significativa (quella che è minore di 256) è contenuta nella locazione di memoria 53266 ($D012 in esadecimale) e, se il bit 7 della locazione 53265 ($D011) è inizializzato a 1, al numero precedente si aggiunge 256 per arrivare al valore corretto che identifica la riga scandita. Naturalmente, dato che questo numero cambia 15780 volte al secondo, un programma BASIC viene eseguito di gran lunga troppo lentamente per poter leggere i registri e poter prendere decisioni significative basate sul loro contenuto. Solo un programma in linguagio macchina ha la velocità per ottenere qualcosa su di una particolare riga di scansione del cursore schermo e addirittura può non essere abbastanza rapido da cambiare grafica senza alcuna leggera, ma visibile, irregolarità.

I registri di cursore schermo hanno due scopi. Quando vengono letti indicano quale riga è attualmente scandita, ma se vengono inizializzati, identificano una particolare riga come il punto in cui una interruzione di cursore schermo è desiderata. Se l'interruzione di cursore schermo viene abilitata, il programma che serve l'interruzione verrà eseguito nel momento esatto in cui il raggio del cursore raggiunge la riga in questione. Ciò consente all'utente di ridefinire ciascuno dei registri del chip VIC-II in qualsiasi punto del video e, in questo modo, cambiare il set di caratteri, il colore di fondo o il tipo di grafica solo per una parte dello schermo.

Realizzare un programma di interruzione del cursore schermo non è, per ammissione generale, lavoro per programmatori principianti, ma, seguendo passo per passo le seguenti spiegazioni, la maggior parte dei programmatori in linguagio macchina dovrebbe essere in grado di scrivere questo genere di sottoprogramma. Coloro che non hanno esperienza di programmi in linguaggio macchina dovrebbero leggere attentamente le spiegazioni, in modo da acquisire un'idea generale di ciò che avviene. Più avanti vedremo come potete usare il sottoprogramma di interruzione presentato come esempio, anche se non sapete nulla sul linguaggio macchina.

Capitolo due

Come si scrive un programma di interruzione cursore schermo

Quando avete terminato di scrivere il sottoprogramma in linguaggio macchina che volete far eseguire al programma che serve l'interruzione, i passi richiesti per realizzare una interruzione del cursore schermo sono i seguenti:

1. Predisporre la variabile di disabilitazione delle interruzioni con una istruzione SEI. Questo disabiliterà tutte le interruzioni ed impedirà al sistema di bloccarsi mentre voi cambiate i vettori di interruzione.
2. Abilitare le interruzioni di cursore schermo. Ciò si ottiene ponendo a 1 il bit 0 del registro di abilitazione delle interruzioni del chip VIC-II alla locazione 53274 ($D01A).
3. Indicare la linea di scansione a cui volete che l'interruzione avvenga, scrivendo nel registro di scansione schermo. Non dimenticate che si tratta di un indirizzo a 9 bit e voi dovete predisporre sia i bit meno significativi (in locazione 53264) sia i più significativi (nel registro alla locazione 53265), in modo da assicurarvi che la interruzione cominci alla riga di scansione desiderata e non 256 righe prima o dopo.
4. Far sapere al calcolatore dove inizia il sottoprogramma in linguaggio macchina che sarà eseguito dal programma che gestisce le interruzioni. Ciò si ottiene posizionando l'indirizzo nel vettore delle interruzioni alle locazioni 788-789 ($314-$315). Questo indirizzo è diviso in due parti: dei bit meno significativi e dei bit più significativi, con i meno significativi immagazzinati in 788. Per calcolare i due valori numerici per un indirizzo dato AD potete usare la formula byte-più-significativo=INT(AD/256) e byte-meno-significativo=AD-(byte-più-significativo * 256). Il valore numerico byte-meno-significativo va in locazione 788 ed il valore byte-più-significativo in 789.
5. Ri-abilitare le interruzioni con una istruzione CLI, che annulla il flag di disabilitazione delle interruzioni del registro di stato.

Quando il computer viene acceso il vettore delle interruzioni è inizializzato in modo da puntare al normale sottoprogramma di servizio delle interruzioni provocate dal temporizzatore hardware, quello che fa avanzare l'orologio al sessantesimo di secondo e legge la tastiera. Poiché questo sottoprogramma di gestione delle interruzioni utilizza lo stesso vettore del sottoprogramma che gesti-

sce le interruzioni del cursore schermo, è preferibile disattivare il temporizzatore hardware, ponendo un valore numerico di 127 nella locazione 56333. Se volete che la tastiera e l'orologio a sessantesimo di secondo funzionino normalmente, mentre il vostro sottoprogramma di gestione delle interruzioni è abilitato, dovete preservare il contenuto delle locazioni 788 e 789 prima di cambiarle per puntare al vostro nuovo sottoprogramma. Quindi dovete fare in modo che il vostro sottoprogramma di gestione delle interruzioni salti al vecchio sottoprogramma di gestione delle interruzione esattamente una volta per ogni scansione dello schermo (ogni sessantesimo di secondo).

Un'altra cosa che dovete tener bene in mente è che almeno due interruzioni cursore schermo sono richieste, se volete cambiare solo una parte dello schermo. Il sottoprogramma di gestione delle interruzioni non deve solo cambiare l'uscita video, ma deve anche produrre un'altra interruzione che la riporti allo stato iniziale.

Il programma 1 è un programma BASIC che utilizza un sottoprogramma di gestione delle interruzioni provocate dalla scansione del cursore schermo per suddividere il video in tre sezioni. Le prime 80 righe di scansione sono in alta risoluzione, formato a mappa di bit, le successive 40 sono in testo normale e le ultime 80 sono in formato a mappa di bit multicolore. Lo schermo si suddividerà nel modo suddetto non appena si verificherà una istruzione SYS al sottoprogramma che attiva le interruzioni, e rimarrà suddiviso anche dopo il termine del programma. Solo se premete insieme i tasti STOP e RESTORE lo schermo ritornerà in condizioni normali.

.Il programma 2 mostra come possa essere organizzata una suddivisione dello schermo completamente diversa, usando lo stesso sottoprogramma in linguaggio macchina. Le istruzioni DATA per il sottoprogramma di gestione delle interruzioni sono le stesse del programma 1, a parte la tabella che inizia a riga 49264. Cambiando questa tabella otteniamo uno schermo che mostra tutti e tre i formati di testo: normale, colore di fondo esteso e multicolore. Testi in maiuscolo ed in minuscolo sono mischiati ed ogni area ha differenti colori di fondo. Questo programma mostra anche come potete cambiare i valori della tabella durante un programma, inserendo il nuovo valore nelle locazoni di memoria dove questa tabella è immagazzinata tramite una istruzione POKE. In questo modo potete, ad esempio, cambiare il colore di fondo di ciascuna delle sezioni

dello schermo mentre il programma è in esecuzione.

Una volta che voi sapete come utilizzare tutte le possibilità grafiche che il VIC-II mette a disposizione, il programma di gestione delle interruzioni qui esemplificato vi dovrebbe mettere in condizione di combinare diversi formati grafici sullo stesso schermo, così che possiate trarre il massimo vantaggio dalla potenza grafica del C64.

Registri di controllo

Le interruzioni usano una tavola di valori che vengono inseriti con delle POKE in quattro locazioni chiave durante ciascuna delle tre interruzioni, come valori per stabilire a quale riga di scansione avvengono le interruzioni. Le locazioni coinvolte sono il registro di controllo 1, il registro di controllo 2, il registro di controllo della memoria, e il registro colore di fondo 0.

Il registro di controllo 1 (nella locazione 53265) permette di scegliere tra il formato di testo colore di fondo esteso, il formato a mappa di bit, l'annullamento dello schermo e 24 o 25 righe di testo. Il registro di controllo 2, in 53270, controlla la selezione del formato multicolore e di una visualizzazione a 38 o 40 colonne. Il registro di controllo della memoria (53272) permette di scegliere quale porzione della memoria utilizzare come gestione video (memoria di schermo) e quale per i dati che definiscono la forma dei caratteri di testo. Il registro colore di fondo 0 (53281) controlla il colore di fondo in formato testo. Informazioni più dettagliate circa l'assegnamento dei bit di queste locazioni si possono reperire nell'Appen dice Ø del *Manuale d'uso del Commodore 64* e nella *Guida di riferimento del programmatore.*

I dati per il sottoprogramma di interruzione sono contenuti nelle righe 49152-49276 del programma 1. Ciascuno di questi numeri di riga corrisponde alle locazioni dove il primo dato del byte contenuto nell'istruzione viene inserito in memoria. Se andate a guardare alle righe 49264-49276 del programma BASIC, troverete delle istruzioni di commento (REM=remarks), che spiegano quale registro del VIC-II ha a che fare con l'istruzione DATA che compare in ciascuna riga. I numeri in queste frasi DATA compaiono in ordine inverso a quello in cui vengono posti nei registri VIC. Ad esempio, la riga 49273 contiene i dati che andranno nel registro di controllo 2. L'ultimo numero, 8, è quello che verrà posto nel registro di controllo 2, mentre viene visualizzata la parte superiore dello schermo.

Il primo numero, 24, viene posto nel registro di controllo 2 durante la visualizzazione della parte inferiore dello schermo e cambia quella parte dello schermo in formato multicolore.

Il solo aspetto complicato nello stabilire la corrispondenza tra istruzioni DATA e interruzione si presenta in riga 49264, la quale contiene i dati che stabiliscono le linee di scansione in cui avvengono le interruzioni. Il dato contenuto in ciascuna istruzione DATA rappresenta la linea di scansione in cui avverrà la *prossima* interruzione. Il primo valore in riga 49264 è 49. Anche se questo dato riguarda la terza interruzione, questo numero corrisponde alla sommità dello schermo (solo le righe di scansione da 50 a 249 sono visibili sullo schermo). Ciò avviene perché dopo la terza interruzione quella successiva ad essere generata è la prima, che avviene alla sommità dello schermo. Analogamente, il valore dell'ultimo dato, 129, viene utilizzato durante la prima interruzione per far partire la successiva alla linea di scansione 129, a metà dello schermo. Provate a fare esperimenti con questi valori per vedere che risultati ottenete. Ad esempio, se cambiate il valore numerico 170 in 210, aumenterete l'area di testo di 5 righe (40 linee di scansione).

Come cambiare il risultato

Cambiando i valori numerici della tavola dati potete modificare il risultato di ciascuna interruzione. Cambiate il valore numerico 20 di riga 49276 in 22, ad esempio, ed otterrete testo in caratteri minuscoli a metà dello schermo. Cambiate il primo 8 di riga 49273 in 24, ed otterrete testo in formato multicolore nella finestra centrale. Ciascuno dei valori di questa tabella può essere usato nello stesso modo in cui usereste i registri corrispondenti, allo scopo di cambiare il colore di fondo, per ottenere grafica in formato di testo o a mappa di bit, in formato standard o multicolore, annullamento dello schermo o colore di fondo esteso.

Programma 1. Testo e grafica

```
10 FORI=49152TO49278:READA:POKEI,A:NEXT:SY
   S12*4096
20 PRINTCHR$(147):FORI=0TO8:PRINT:NEXT
```

```
30 PRINT"L'AREA SUPERIORE E' IN ALTA RISOL
   UZIONE,FORMATO A MAPPA DI BIT; ";
40 PRINT"QUELLA CENTRALE IN FORMATO CARATT
   ERE STANDARD; L'INFERIORE IN FORMATO";
50 PRINT" MULTICOLORE A MAPPA DI BIT"
60 FORG=1024TO1383:POKEG,114:NEXT:FORG=138
   4TO1423:POKEG,6:NEXT
70 FORG=1664TO2023:POKEG,234:NEXT
80 FORG=55936TO56295:POKEG,13:NEXT
90 FORI=8192TO11391:POKEI,0:POKEI+4800,0:N
   EXT
100 BASE=2*4096:BK=49267
110 H=40:C=0:FORX=0TO319:GOSUB150:NEXT
120 H=160:C=0:FORX=0TO319STEP2:GOSUB150:NE
    XT:C=40:FORX=1TO319STEP2:GOSUB150:NEXT

130 C=80:FORX=0TO319STEP2:W=0:GOSUB150:W=1
    :GOSUB150:NEXT
140 GOTO140
150 Y=INT(H+20*SIN(X/10+C)):CH=INT(X/8):RO
    =INT(Y/8):LN=YAND7
160 BY=BASE+RO*320+8*CH+LN:BI=ABS(7-(XAND7
    )-W)
170 POKEBY,PEEK(BY)OR(2↑BI):RETURN
49152 DATA120,169,127,141,13,220
49158 DATA169,1,141,26,208,169
49164 DATA3,133,251,173,112,192
49170 DATA141,18,208,169,24,141
49176 DATA17,208,173,20,3,141
49182 DATA110,192,173,21,3,141
49188 DATA111,192,169,50,141,20
49194 DATA3,169,192,141,21,3
49200 DATA88,96,173,25,208,141
49206 DATA25,208,41,1,240,43
49212 DATA198,251,16,4,169,2
49218 DATA133,251,166,251,189,115
49224 DATA192,141,33,208,189,118
49230 DATA192,141,17,208,189,121
49236 DATA192,141,22,208,189,124
```

```
49242 DATA192,141,24,208,189,112
49248 DATA192,141,18,208,138,240
49254 DATA6,104,168,104,170,104
49260 DATA64,76,49,234
49264 DATA49,170,129 :REM SCANDISCE LE RIG
      HE
49267 DATA0,6,0 :REM COLORI DI FONDO ESTES
      O
49270 DATA59,27,59 :REM REGISTRO DI CONTRO
      LLO # 1
49273 DATA24,8,8 :REM REGISTRO DI CONTROLL
      O # 2
49276 DATA24,20,24 :REM CONTROLLORE MEMORI
      A
```

Programma 2. I tre formati di testo

```
10 FORI=49152TO49278:READA:POKEI,A:NEXT:SY
   S12*4096
20 PRINTCHR$(147)CHR$(5):POKE53280,0
30 POKE53280,0:POKE53282,6:POKE53283,5:POK
   E53284,4
40 PRINT:PRINT"QUESTO E' IL FORMATO MULTIC
   OLORE."
50 PRINT:PRINT"I CARATTERI A QUATTRO COLOR
   I SONO DIFFI-CILI DA LEGGERE."
60 PRINT:PRINTCHR$(150)"{ 2 SPAZI}ABCDEFGH
   IJKLMNOPQRTUVWXYZ1234567890"
70 PRINT:PRINT:PRINT:PRINTCHR$(28)"QUESTO
   E' IL FORMATO STANDARD...."
80 PRINT:PRINT"{ 6 SPAZI}QUI LA FANTASIA N
   ON C'ENTRA!":PRINT:PRINT:PRINT
90 PRINTCHR$(144)"{ 5 SPAZI}{RVS}FOR{OFF}M
   A{RVS}TO{OFF} CO{RVS}LO{OFF}RE {RVS}DI
   {OFF} {RVS}FO{OFF}ND{RVS}O{OFF} {RVS}ES
   {OFF}TE{RVS}SO{OFF}"
100 PRINT:PRINT"{ 4 SPAZI}USATE DIFFERENTI
     COLORI DI FONDO"
110 PRINT"{ 4 SPAZI}{RVS}USATE DIFFERENTI
    COLORI DI FONDO"
```

```
120 PRINT"{ 4 SPAZI}USATE{SPAZI}DI{ 2 F}
    ERENTI{SPAZI}COLORI{SPAZI}DI{SPAZI}FO
    NDO"
130 PRINT"{ 4 SPAZI}{RVS}USATE{SPAZI}DI
    { 2 F}ERENTI{SPAZI}COLORI{SPAZI}DI
    {SPAZI}FONDO"
140 FORS=0TO3000:NEXT
150 FORS=49267TO49269:POKES,RND(1)*16:FORI
    =1TO2000:NEXTI,S:GOTO140
49152 DATA120,169,127,141,13,220
49158 DATA169,1,141,26,208,169
49164 DATA3,133,251,173,112,192
49170 DATA141,18,208,169,24,141
49176 DATA17,208,173,20,3,141
49182 DATA110,192,173,21,3,141
49188 DATA111,192,169,50,141,20
49194 DATA3,169,192,141,21,3
49200 DATA88,96,173,25,208,141
49206 DATA25,208,41,1,240,43
49212 DATA198,251,16,4,169,2
49218 DATA133,251,166,251,189,115
49224 DATA192,141,33,208,189,118
49230 DATA192,141,17,208,189,121
49236 DATA192,141,22,208,189,124
49242 DATA192,141,24,208,189,112
49248 DATA192,141,18,208,138,240
49254 DATA6,104,168,104,170,104
49260 DATA64,76,49,234
49264 DATA49,177,113 :REM SCANDISCE LE RIG
      HE
49267 DATA2,7,6 :REM COLORI DI FONDO ESTES
      O
49270 DATA91,27,27 :REM REGISTRO DI CONTRO
      LLO # 1
49273 DATA8,8,24 :REM REGISTRO DI CONTROLL
      O # 2
49276 DATA20,22,20 :REM CONTROLLORE MEMORI
      A
```

Blocco per schizzi ad alta risoluzione

Chris Metcalf

I disegni ad alta risoluzione possono essere ricchi di dettagli e spettacolari. Peraltro crearli può essere difficile. Il «blocco per schizzi ad alta risoluzione» facilita il compito di creare grafica ad alta risoluzione. Una volta che avete creato i vostri capolavori, è facile registrarli su disco o su nastro per un uso futuro nei vostri programmi.

Le magiche parole *alta risoluzione* sono uno degli argomenti che mi hanno spinto a comprare un Commodore 64. Senza dubbio anche voi sarete stati influenzati dall'idea di avere una mappa di 320x200 punti sullo schermo, un totale di 16 colori da stendere sullo schermo e la possibilità di mescolare fino a quattro colori all'interno di ciascuna area di 8x8 pixel.

Sfortunatamente, è molto difficile utilizzare queste potenti opzioni. Al Commodore 64 mancano dei comandi BASIC per gestire l'alta risoluzione (come i comandi BASIC PLOT, POSITION, DRAWTO e LOCATE dell'Atari), ma esso dispone in effetti di un paio di formati a mappa di bit ad alta risoluzione con grosse possibilità. La sola difficoltà consiste nell'accedervi in BASIC.

Il BASIC fornisce un minimo controllo sulla grafica. È necessaria una serie di istruzioni POKE persino per attivare lo schermo ad alta risoluzione, quindi altre POKE sono necessarie per cancellare la pagina grafica per poterla usare. Una volta che questo è stato ottenuto, altre POKE sono richieste per tracciare punti sullo schermo e per attribuire loro i colori voluti. Questo procedimento è lento, noioso e difficile.

La grafica ad alta risoluzione

In altre parti di questo libro si possono reperire descrizioni dettagliate dei formati grafici ad alta risoluzione, ma un breve riassunto può essere utile. Lo schermo a mappa di bit può essere fisicamente

sistemato in una qualsiasi delle otto aree da 8Kbyte della memoria. Il programma «blocco per schizzi» utilizza per questo schermo le locazioni da 40960 a 48959 ($A000-$BF3F). I dati riguardanti i colori sono registrati in altra parte della memoria. Nel formato standard ad alta risoluzione il colore può essere ricavato da uno qualsiasi dei blocchi da 1Kbyte nella stessa area di 16Kbyte di memoria dello schermo a mappa di bit. Il blocco per schizzi usa l'area da 35840 a 36839 ($8C00-$8FE7) per questa memoria colore mobile. Nel formato a mappa di bit multicolore è necessario altro spazio in memoria per ospitare i colori aggiunti; questa memoria colore è fissata nell'area da 55296 a 56296 ($D800-$DB87).

Sul C64 lo schermo ad alta risoluzione simula, nel suo formato, 1000 caratteri programmabili. Il primo byte dello schermo definisce gli otto pixel all'inizio della riga superiore. I sette byte successivi definiscono i primi otto pixel di ogni riga seguente. Tuttavia, il gruppo successivo di otto byte non è posto al di sotto, ma a destra dei primi otto. Dopo 40 gruppi di otto byte (l'equivalente di una riga di caratteri programmabili) la sequenza viene ripetuta per le successive otto righe di pixel.

Nel formato ad alta risoluzione normale sia il colore di fondo che quello dei pixel sono definiti dall'area selezionabile da 1Kbyte di memoria colore. Il nybble (quattro bit, mezzo byte) più significativo definisce il colore di tutti i pixel all'interno di un gruppo di 8x8 pixel (un «carattere»). Il nybble meno significativo definisce il colore di fondo della stessa area (visibile quando il bit è nullo).

Il formato multicolore consente più colori all'interno di un'area di 8x8 pixel, assegnando uno dei quattro colori possibili ad ogni combinazione di due bit. Tuttavia il risultato è che sono necessari entrambi i bit per definire un singolo pixel. Ciascuna coppia di bit ricava il suo colore dal corrispondente byte dello schermo colore mobile, dello schermo fisso o del registro colore di fondo, nel modo seguente:

Coppie di bit	Sorgente del codice colore
00	registro colore di fondo (53280, $D021)
01	nybble più significativo della memoria colore mobile
10	nyble meno significativo della memoria

11 — colore mobile
memoria colore fissa

Come nel formato ad alta risoluzione normale, la memoria colore fornisce informazioni separate riguardanti il colore a ciascun gruppo di 8x8 pixel. Tuttavia, contrariamente al formato standard ad alta risoluzione, tutti i bit 00 sono inizializzati a partire dal registro 53281.

Nondimeno il blocco per schizzi ad alta risoluzione vi consente di ignorare la maggior parte di questi dettagli. Comunque avrete già capito per quale ragione non potete utilizzare troppi colori contemporaneamente: semplicemente, i nuovi colori cambiano il colore dei pixel opportuni all'interno di ciascuna area 8x8.

Come copiare il blocco per schizzi

Il blocco per schizzi è progettato per essere usato con programmi BASIC residenti in memoria contemporaneamente. Il programma stesso parte in 36864 ed arriva fino a 40095 ($9000-$9C9F); diverse tavole di dati vanno da 40192 a 40959 ($9D00-$9FFF) e da 51968 a 52223 ($CB00-$CBFF). Lo schermo colore mobile si trova in 35840, quindi il limite superiore del BASIC è posto in 35839 dal blocco per schizzi. L'area di schermo a mappa di bit va da 40960 a 48959. Normalmente l'interprete BASIC occupa quest'area di memoria, ma una istruzione POKE 1,54 la rende disponibile ad altri usi. Tuttavia, questa istruzione POKE non può essere usata direttamente da BASIC (dal momento che l'interprete non sarebbe più presente) e può essere utilizzata solo in linguaggio macchina.

Questo programma è stato interamente scritto in linguaggio macchina, quindi è necessario copiarlo usando un programma che consente di copiare testi in linguaggio macchina (MLX — Appendice I).

L'uso del programma MLX renderà molto più semplice l'introduzione dei programmi in linguaggio macchina. Leggete attentamente e comprendete a fondo le istruzioni per l'uso dell'MLX prima di tentare di copiare il blocco per schizzi. Le informazioni richieste per inserire il blocco per schizzi ad alta risoluzione tramite MLX sono:

indirizzo iniziale: 36864
indirizzo finale: 40095.

Quando avete terminato di copiarlo usate il comando Save dell'MLX, in modo da creare una copia del vostro programma su disco o nastro.

Tutte le volte che volete usare il programma battete LOAD «SKETCHPAD», 8, 1 per i dischi o LOAD «SKETCHPAD», 1, 1 per i nastri. Per far partire il programma battete SYS 36864 e premete RETURN. Dovrebbe apparire il seguente messaggio:

HIRES SKETCHPAD — BY CHRIS METCALF
MULTICOLOR MODE?N

ed il cursore lampeggerà sulla N. A questo punto premete Y(si) o N (no) per stabilire se userete il formato a mappa di bit normale o multicolore nel corso del programma. Se non battete nulla, il programma si interrompe. Il formato normale fornisce una miglior risoluzione per disegni più complessi, mentre quello multicolore è più utile per disegni meno dettagliati e più ricchi di colori.

Semplici grafici con il blocco per schizzi

Una volta premuto RETURN dovrebbe apparire lo schermo a mappa di bit. Un piccolo sprite a forma di tartaruga nel centro dello schermo indica il punto in cui state disegnando. La prima volta che caricate il programma, dopo aver acceso il calcolatore, lo schermo sarà coperto di pixel colorati a caso. Premete SHIFT-CLR per cancellarlo. In qualsiasi momento potete premere CTRL ← (frecciolina a sinistra) per interrompere il programma.

Il programma è stato progettato per poter controllare la tartaruga sia con il joystick che con la tastiera. Gli utenti in possesso del joystick possono muovere la tartaruga con il joystick inserito nella Control Port 2 e possono controllare i vari formati con il pulsante di sparo. Tuttavia, è stato definito anche un certo numero di tasti per poter spostare la tartaruga. I quattro tasti Q, E, Z e C, agli angoli, permetteranno di muovere la tartaruga in tutte le otto direzioni.

in alto/ a sinistra	**in alto**	**in alto/ a destra**
Q	**W**	**E**

a sinistra A	S	D a destra
Z	X	C
a sinistra/ in basso	in basso	a destra/ in basso

Il tasto S al centro del riquadro viene usato per far tornare la tartaruga alla sua posizione di partenza al centro dello schermo ed il tasto HOME pone la tartaruga nell'angolo superiore sinistro dello schermo.

Il primo esperimento da compiere consiste nel limitarsi a tracciare qualche linea. Per attivare il formato grafico premete la barra spaziatrice (o SHIFT-barra spaziatrice) o il pulsante di sparo del joystick. Apparirà un punto al centro della tartaruga. Ora traccerete una linea in qualsiasi direzione vi muoviate. Per smettere di dise gnare e potervi muovere semplicemente premete un'altra volta la barra spaziatrice o il pulsante di sparo.

Quando caricate per la prima volta il programma la tartaruga avrà lo stesso colore che stavate usando (il colore del carattere). Per cambiare questo colore usate i tasti CTRL o Commodore con un tasto numerico da 1 a 8. I colori ottenibili con il tasto CTRL appaiono sulla faccia anteriore dei tasti numerici. Il colore che usate viene tracciato con qualsiasi punto disegnate; quindi, se cercate di disegnare in un'area di 8x8 pixel in cui avete già disegnato con un colore differente, il colore dell'area di pixel cambierà nel colore attuale del vostro punto.

Il formato multicolore

Il problema suddetto può essere ridotto usando il formato multicolore. Quando attivate il programma con una SYS36864, battete Y per multicolore. Vi accorgerete che i pixel che tracciate sono in pratica più larghi. Tuttavia, potete ora mischiare liberamente i colori. Ciascuno dei tre tipi di tracciato (le coppie di bit 01, 10 e 11) ed il formato di cancellatura (00) sono rappresentati dai tasti funzione. Il tasto f1 corrisponde a 11, f3 a 10 e f5 a 01. Quando il programma inizia partite con f5. Usando un qualsiasi tipo di tracciato avete le stesse limitazioni con i colori che affliggono il formato a mappa di bit normale. Tuttavia, la colorazione di uno qualunque dei tre tipi è completamente indipendente, quindi alternando f1, f3

e f5 potete tracciare linee senza alterare i colori degli altri tipi di tracciato. Il tasto f7 vi porrà nel formato che vi consente di cancellare.

Nel formato a mappa di bit normale possono essere usati gli stessi tasti funzione. In entrambi i formati possono anche essere utilizzati i tasti più e meno (anche accoppiati allo SHIFT), che corrispondono, rispettivamente, a f5 ed f7. Il tracciato normale in alta risoluzione standard è con f5, ed è con esso che iniziate. Il tasto f1 è stato predisposto per abbandonare f5 quando viene premuto in formato a mappa di bit normale. Il formato f7 ha la stessa funzione che in formato multicolore: cancella i pixel senza influenzare il colore dei pixel adiacenti. Il formato f3 non traccia alcun pixel, cambia invece il colore di fondo all'interno del riquadro 8x8, senza alterare i colori dei pixel già tracciati. Tracciare linee in questo formato può essere un buon modo per familiarizzare con l'organizzazione 8x8 pixel a colori.

Opzioni speciali

Si può anche cambiare colore di cornice e di fondo all'interno di questo programma. Tuttavia, se vi trovate in formato a mappa di bit normale, il colore di fondo della mappa di bit non cambierà finché non premerete SHIFT-CRL. I colori di fondo e della cornice vengono cambiati con il joystick o con i tasti direzionali. Per entrare nel formato che vi consente di cambiare colore premete il tasto ↑ (frecciolina verso l'alto). Muovendo il cursore a destra e a sinistra, od usando i corrispondenti tasti, cambierà il colore del bordo. Per cambiare il colore di fondo (il cambiamento sarà immediatamente evidente solo in formato multicolore) spostate il joystick verso l'alto o il basso, oppure usate i tasti. Per uscire da questa opzione premete il pulsante di sparo od un tasto qualsiasi, che non sia uno di quelli direzionali, e ritornerete nel ciclo principale.

Il movimento passo dopo passo è un'altra delle opzioni di questo programma. Quando iniziate voi muovete un pixel alla volta. Tuttavia, ogni volta che premete un tasto numerico (anche accoppiato allo SHIFT) comincerete a muovervi di altrettanti pixel per volta. Ad esempio, se volete tracciare linee a velocità doppia, premete il tasto 2; per spostarvi di otto pixel per volta, premete il tasto 8. La stessa opzione funziona in formato multicolore, ma, a causa dei pixel di larghezza doppia, i numeri dispari danno talvolta risultati piuttosto bizzarri.

Capitolo due

Formati grafici più evoluti

Premendo i tasti funzione ed il tasto SHIFT sono disponibili opzioni ancora più potenti. La prima, nota come formato disegna-da, viene attivata e disattivata da f2. Quando premete f2 il sottoprogramma che traccia le linee assume la posizione attuale come punto di partenza. Ora, mentre vi muovete per lo schermo, vedrete una linea che congiunge la vostra tartaruga con il punto di partenza scelto. Questo elastico non cambia i pixel adiacenti. Tuttavia cambia i colori, se vi trovate in uno dei formati di tracciamento. Solo con f7 o con '-' non verranno tracciati colori dalla linea elastica mentre vi muovete. Una volta che la linea è in una posizione che vi soddisfa, premete il tasto SHIFT, od il pulsante di sparo, ed una linea verrà effettivamente tracciata nel colore e nel formato di tracciamento in cui vi trovate.

Mentre continuate a muovervi ed a tracciare linee, il punto di partenza rimarrà dove lo avete fissato. Ciò vi permette di creare complicate opere astratte scegliendo semplicemente una spaziatura tripla o quadrupla (o qualsiasi altra desideriate) e spostandovi lungo lo schermo tenendo premuto il tasto SHIFT od il pulsante di sparo. Si può usare anche il tasto SHIFT-LOCK. Tuttavia, la barra spaziatrice continuerà ad attivare ed a disattivare il tracciato semplice al di sotto della tartaruga. Per interrompere il formato disegna-da premete nuovamente f2. Quindi, per assegnare la vostra posizione attuale come nuovo punto di partenza premete nuovamente f2. Notate che dal momento in cui il tasto SHIFT traccerà una linea è spesso utile usare il tasto con il marchio Commodore per caratteri cui si applica semplicemente il tasto SHIFT (per esempio, SHIFT-CRL, SHIFT-f8), dato che ottiene gli stessi risultati.

Il secondo formato viene selezionato con il tasto f4. Questo è il formato disegna-fino-a, che è molto simile a quello disegna-da. Tuttavia, in questo formato, ogni volta che premete il pulsante di sparo od il tasto SHIFT viene tracciata una linea e la vostra posizione attuale assunta come nuovo punto di partenza. Ciò produce lo stesso effetto del comando DRAWTO di Atari. Il formato disegna-fino-a vi permette di tracciare molto facilmente figure geometriche. Notate che, se vi trovate in formato f4 e scegliete f2, f4 verrà annullato e sostituito da f2. Anche la transazione opposta è legittima.

Il terzo formato, che consente di tracciare linee (f6), è soprattutto utile per realizzare figure ombreggiate. Quando questo formato è

attivo, tutte le volte che premete il tasto SHIFT od il pulsante di sparo verrà tracciata una linea verso destra fino a che non incontra un altro pixel attivo od il margine destro. Il formato non ha alcun effetto elastico.

Potete anche fissare il termine della linea tracciata dalla opzione precedente. Normalmente la linea si interrompe quando incontra un pixel tracciato nello stesso formato, quindi cancellerebbe un'area piena sulla destra o continuerebbe in formato f3 fino a che non incontrasse una coppia di bit f3, e così via, in base al formato corrente. Tuttavia, l'asterisco attiverà un cambiamento. Se premete il tasto '*' una sola volta, dopo l'inizio del programma, il formato di tracciamento andrà alla ricerca di qualsiasi pixel attivato. In questo modo potete tracciare una linea verso destra in formato f1 e fermarvi in corrispondenza di un pixel tracciato in formato f3, creando una cornice di colore differente. Premendo nuovamente il tasto si ritornerà al formato iniziale, che si arresta sui pixel di egual tipo.

Campitura

Una delle opzioni più potenti del programma viene richiamata quando premete il tasto f8. Questo tasto attiva la campitura. Questa funzione riempirà ogni area racchiusa fra pixel o compresa tra una linea continua di pixel e la cornice. Fino a che punto campisce dipende anche dal tasto asterisco. Normalmente campisce fino ad incontrare un qualsiasi pixel dello stesso tipo, ma gli si può segnalare di campire fino ad un qualsiasi pixel attivato, permettendo cosi di creare cornici di colore differente in formato multicolore. La campitura sfuggirà da qualsiasi figura non completamente chiusa, ma non se racchiusa da pixel collegati diagonalmente.

La riga di stato

Tutti questi formati sono piuttosto difficili da ricordare. Dopo tutto, con quattro formati di tracciamento, una opzione traccia mento/cancellatura, tre tipi di linee ed un tipo di campitura le idee possono farsi piuttosto confuse. Specialmente se si pensa che l'opzione traccia-fino-a non ha alcun effetto elastico, il formato attivato dal tasto '-' e quello di cancellatura hanno lo stesso aspetto e le linee tracciate in multicolore sono indistinguibili quando sono dello stesso colore. Per aiutarvi a ricordarli si può attivare una riga di stato ai piedi dello schermo, premendo e tenendo abbassato il tasto RETURN.

La riga di stato è composta da quattro parti. La prima indica il formato in cui vi trovate (f1, f3, f5 o f7). La seconda segnala se state tracciando una linea o se vi state solo muovendo per lo schermo. La terza parte mostra il formato di tracciamento linee in cui vi trovate (OFF, FROM, TO o LINE) e la quarta comunica lo stato del formato asterisco (si comincia con SAME; ANY significa campitura interrotta da qualsiasi pixel attivo).

Ingresso/uscita per il blocco per schizzi

Il programma è fornito di una opzione che consente di caricare e registrare i dati che compongono un'immagine in alta risoluzione. Per attivare questa opzione premete il tasto @. Il programma vi chiederà se volete caricare (LOAD) o registrare (SAVE): la sola prima lettera è significativa (L o S). Qualsiasi altra risposta interromperà il procedimento. Quindi dovete specificare il numero della periferica. Il registratore è 1 e l'unità dischi può essere sia 8 (come nella maggior parte dei casi) che 9 (si può anche usare la periferica identificata dal numero 2, il canale RS-232, ma sarebbero necessarie modifiche al linguaggio macchina per inserire le specifiche riguardanti il livello di baud ed altri parametri). Non sono consentite altre periferiche. Infine dovete fornire un nome. Se non viene specificato alcun nome, il procedimento ha termine. La tartaruga sparirà per il tempo necessario all'operazione di LOAD o SAVE. Quando il procedimento è terminato la tartaruga riapparirà.

Le operazioni di ingresso/uscita da disco sono semplici. Basta specificare L o S, 8 ed il nome. Assicuratevi che ci sia un disco nell'unità disco e, particolarmente importante, accendetela e collegatevi. *Se il disco non è pronto, il programma si arresterà.* In questo caso tutto ciò che potete fare è premere RUN/STOP-RESTORE. Non sono necessari suffissi per la registrazione od il caricamento da disco, ma tutti i prefissi voluti (come «0:» o «@0:») devono essere inclusi nel nome. Quando l'unità disco ha terminato il canale di errore viene letto e mostrato per un paio di secondi. Normalmente si ottiene un «00,OK,00,00). Alcune delle segnalazioni di errore più diffuse sono:

62, FILE NOT FOUND	— Si è cercato di caricare un file inesistente
63, FILE EXISTS	— Registrate sotto un nuovo nome o con

	«@0:».
64, FILE TYPE MISMATCH	— Si registra con «@0:» su di un file di programma.
72, DISK FULL	— Procuratevi un altro disco o cancellate dei file.
74, DRIVE NOT READY	— Il canale dell'unità dischi è aperto; registrate con «0:».

Qualsiasi altro errore (in particolare il 21) indica qualche irregolarità di funzionamento del disco. Fate riferimento al manuale dell'unità 1541.

I possessori di unità nastro non hanno a che fare con le segnalazioni d'errore. Per registrare (SAVE) o caricare (LOAD) con l'unità nastro battete L o S, 1 ed il nome. Tuttavia è consigliabile premere PLAY o RECORD & PLAY prima di premere RETURN per l'ultima domanda. In questo modo il registratore non invierà alcun messaggio. I messaggi provocano la sovrapposizione di colori indesiderati sui colori prefissati dello schermo. Inoltre, se il messaggio provoca lo scorrimento dello schermo, anche lo schermo colore scorrerà con esso e cancellerà tutte le informazioni sul colore f1 multicolore (11). Tuttavia, persino questo fatto non ha effetti catastrofici. Per impedirlo, limitatevi a cancellare lo schermo prima di battere SYS 36864, per premunirvi dai messaggi di errore. Potete premere RUN/STOP durante la registrazione od il caricamento e ritornare direttamente al programma blocco per schizzi.

Le informazioni riguardanti l'alta risoluzione vengono registrate in un formato assolutamente particolare. I primi due byte registrati rappresentano il colore della cornice e del bordo. Sono seguiti dai dati dello schermo colore mobile (1000 byte), dai dati dello schermo colore fisso (altri 1000 byte) e, infine, dallo schermo ad alta risoluzione. Lo schermo viene registrato tramite una tecnica di compattazione dei dati. Tutti i byte non nulli vengono segnalati normalmente, ma uno zero identifica un formato speciale: i due byte successivi rappresentano l'indirizzo del byte successivo non nullo nel formato byte meno significativo, byte più significativo. Questo permette al programma di cancellare rapidamente lo spazio intercorrente e di caricare solo i dati significativi del disegno.

Capitolo due

Sottoprogramma di caricamento/registrazione

Il programma 2 è un sottoprogramma che permette di integrare i disegni del blocco per schizzi nei vostri programmi. Il sottoprogramma è diviso in tre parti principali: il caricatore dei dati, il sottoprogramma stesso (in riga 50000) ed i dati del linguaggio macchina. Il caricatore dei dati va posto all'inizio del vostro programma e si limita a leggere le istruzioni DATA ed a porle in memoria da 51676 a 51976 ($C9DC-$CAFF). Il sottoprogramma tratta le vostre richieste e richiama il linguaggio macchina.

Per usare il sottoprogramma inserite in LS un segnale di caricamento o registrazione (caricamento=0, registrazione=1), in DV il numero che contraddistingue la periferica (8 per il disco, 1 per il nastro) ed in NM$ il nome del file. Quindi ponete l'istruzione GOSUB50000. Il sottoprogramma BASIC, tuttavia, non è necessario; si può richiamare direttamente il linguaggio macchina. Per farlo, inserite, con una POKE, 0 per segnalare il caricamento o 1 per segnalare la registrazione nella locazione 2. Quindi aprite il tipo di file opportuno:

caricamento da disco:
    OPEN 1,8,2, «nome del file»
registrazione su disco:
    OPEN 1,8,2,«nome del file, S.W»
caricamento da nastro:
    OPEN 1 oppure OPEN 1,1,0,«nome»
registrazione su nastro:
    OPEN 1,1,1,«nome del file»

Infine battete SYS 51676. Ad esempio, per caricare un disegno (chiamato «DISEGNO3») da disco:

```
POKE 2,0: OPEN 1,8,2,"DISEGNO3": SYS 51676
```

Linguaggio macchina

Il programma 3 è il codice sorgente per il blocco per schizzi. Il programma può essere copiato usando un assemblatore.

Il codice sorgente è commentato e viene fornito per coloro che sono interessati nello studio del funzionamento del programma. Riportiamo qui sotto una lista degli indirizzi iniziali dei sottopro-

grammi principali.

$9000	inizializzazione; richiamata solo all'inizio del programma.
$9167	ciclo principale - inserimento dati da tastiera.
$93BC	- inserimento dati da joystick.
$945E	- movimento e tracciamento.
$9538	sottoprogramma di tracciamento linee.
$96CE	sottoprogramma di campitura delle aree.
$97C6	sottoprogrammi vari; interruzione cursore video.
$992A	sottoprogramma di caricamento e registrazione.
$9BA5	dati.

Programma 1. Blocco per schizzi ad alta risoluzione

```
36864 :032,231,255,160,000,185,095
36870 :201,155,240,006,032,210,082
36876 :255,200,208,245,160,000,056
36882 :032,207,255,201,013,208,166
36888 :001,096,201,089,208,001,108
36894 :200,152,072,032,207,255,180
36900 :201,013,208,249,160,045,144
36906 :185,057,000,153,000,203,128
36912 :136,016,247,104,133,060,232
36918 :169,000,133,157,169,140,054
36924 :133,056,133,052,169,000,091
36930 :133,055,133,051,169,128,223
36936 :141,138,002,169,197,141,092
36942 :000,221,169,054,133,001,144
36948 :169,056,133,076,169,059,234
36954 :133,075,169,008,164,060,187
36960 :240,002,169,024,133,077,229
36966 :133,078,169,000,133,064,167
36972 :133,057,133,058,133,079,189
36978 :133,080,133,068,169,160,089
36984 :133,067,169,100,133,069,023
36990 :169,001,133,062,133,059,171
36996 :165,060,010,010,010,133,008
37002 :070,169,007,056,229,060,217
37008 :133,073,169,001,024,101,133
37014 :060,133,074,169,140,141,099
```

```
37020  :096,203,169,000,141,064,061
37026  :203,169,160,141,160,203,174
37032  :169,000,141,128,203,170,211
37038  :189,064,203,024,105,040,031
37044  :157,065,203,189,096,203,069
37050  :105,000,157,097,203,189,169
37056  :128,203,024,105,064,157,105
37062  :129,203,189,160,203,105,163
37068  :001,157,161,203,232,224,158
37074  :024,208,217,169,001,160,221
37080  :007,153,192,203,010,136,149
37086  :016,249,169,001,160,006,055
37092  :153,200,203,010,153,208,131
37098  :203,010,136,136,016,244,211
37104  :169,003,160,006,153,216,179
37110  :203,010,010,136,136,016,245
37116  :247,169,254,160,007,153,218
37122  :224,203,056,042,136,016,167
37128  :248,169,252,160,007,153,229
37134  :231,203,153,239,203,153,172
37140  :247,203,056,042,056,042,154
37146  :136,136,016,239,160,040,241
37152  :162,040,165,060,240,002,189
37158  :162,081,189,076,156,153,087
37164  :192,191,202,136,016,246,003
37170  :169,000,160,021,153,233,018
37176  :191,136,016,250,169,255,049
37182  :141,248,143,169,172,024,191
37188  :101,060,141,000,208,169,235
37194  :143,141,001,208,169,000,224
37200  :141,027,208,141,028,208,065
37206  :141,029,208,141,023,208,068
37212  :173,134,002,133,061,141,224
37218  :039,208,032,125,152,169,055
37224  :001,141,139,002,032,228,135
37230  :255,072,165,058,240,040,172
37236  :173,141,002,041,001,208,170
37242  :007,173,000,220,041,016,067
37248  :208,006,032,056,149,076,143
```

```
37254  :156,145,165,058,201,003,094
37260  :240,014,169,001,133,079,008
37266  :032,056,149,032,056,149,108
37272  :169,000,133,079,104,208,077
37278  :003,076,188,147,072,032,164
37284  :021,152,104,164,066,240,143
37290  :003,076,217,147,201,032,078
37296  :240,004,201,160,208,009,230
37302  :165,057,073,001,133,057,156
37308  :076,188,147,201,083,208,067
37314  :015,169,000,133,068,169,236
37320  :160,133,067,169,100,133,194
37326  :069,076,188,147,201,019,138
37332  :208,011,169,000,133,067,032
37338  :133,068,133,069,076,188,117
37344  :147,201,043,240,004,201,036
37350  :219,208,011,169,001,133,203
37356  :059,169,008,133,070,076,239
37362  :188,147,201,045,240,004,043
37368  :201,221,208,005,169,000,028
37374  :076,235,145,201,140,208,235
37380  :006,032,206,150,076,188,150
37386  :147,201,137,208,023,165,123
37392  :058,041,001,073,001,133,067
37398  :058,165,067,133,081,165,179
37404  :068,133,082,165,069,133,166
37410  :083,076,094,148,201,138,006
37416  :208,011,165,058,041,002,013
37422  :073,002,133,058,076,023,155
37428  :146,201,139,208,018,165,161
37434  :058,208,007,169,003,133,124
37440  :058,076,188,147,169,000,190
37446  :133,058,076,188,147,201,105
37452  :042,208,009,165,080,073,141
37458  :001,133,080,076,188,147,195
37464  :201,094,208,010,036,197,066
37470  :080,252,032,198,151,076,115
37476  :103,145,201,013,208,091,093
37482  :032,055,152,160,039,185,217
```

```
37488 :000,156,153,192,143,136,124
37494 :016,247,164,059,185,040,061
37500 :156,141,198,143,165,057,216
37506 :010,010,168,162,000,185,153
37512 :044,156,157,200,143,200,012
37518 :232,224,004,208,244,165,195
37524 :058,010,010,168,162,000,044
37530 :185,052,156,157,215,143,038
37536 :200,232,224,004,208,244,248
37542 :165,080,010,010,168,162,249
37548 :000,185,068,156,157,228,198
37554 :143,200,232,224,004,208,165
37560 :244,165,197,201,001,240,208
37566 :250,032,095,152,076,188,215
37572 :147,201,147,208,072,160,107
37578 :000,132,253,152,162,160,037
37584 :134,254,145,253,200,208,122
37590 :251,232,224,191,208,244,028
37596 :134,254,145,253,200,192,118
37602 :064,208,249,160,000,132,015
37608 :253,165,061,010,010,010,229
37614 :010,077,033,208,041,240,079
37620 :077,033,208,162,140,134,230
37626 :254,145,253,200,208,251,025
37632 :232,224,143,208,244,134,161
37638 :254,145,253,200,192,232,002
37644 :208,249,076,188,147,201,057
37650 :018,208,039,160,000,132,063
37656 :251,169,160,133,252,177,142
37662 :251,073,255,145,251,200,181
37668 :208,247,230,252,165,252,110
37674 :201,191,208,239,177,251,029
37680 :073,255,145,251,200,192,140
37686 :064,208,245,076,188,147,214
37692 :201,006,208,049,169,027,208
37698 :141,017,208,169,021,141,251
37704 :024,208,169,008,141,022,132
37710 :208,169,000,141,021,208,057
37716 :169,199,141,000,221,169,215
```

```
37722  :055,133,001,032,178,152,129
37728  :160,045,185,000,203,153,074
37734  :057,000,169,000,153,000,225
37740  :002,136,016,242,096,201,033
37746  :064,208,003,076,042,153,148
37752  :162,015,221,181,155,208,038
37758  :008,134,061,142,039,208,206
37764  :076,188,147,202,016,240,233
37770  :162,003,221,197,155,208,060
37776  :027,165,060,208,006,224,066
37782  :003,208,002,162,001,134,148
37788  :059,162,008,165,059,240,081
37794  :004,010,010,010,170,134,244
37800  :070,076,188,147,202,016,099
37806  :221,056,041,239,233,032,228
37812  :240,006,201,010,176,002,047
37818  :133,062,165,162,197,065,202
37824  :208,003,076,094,148,165,118
37830  :162,133,065,173,000,220,183
37836  :073,127,133,066,165,197,197
37842  :201,005,208,003,076,094,029
37848  :148,165,066,041,016,240,124
37854  :025,165,058,240,006,032,236
37860  :056,149,076,248,147,165,045
37866  :064,208,015,230,064,165,212
37872  :057,073,001,133,057,076,125
37878  :252,147,169,000,133,064,243
37884  :165,066,041,001,240,011,008
37890  :165,069,056,229,062,201,016
37896  :200,176,002,133,069,165,241
37902  :066,041,002,240,011,165,027
37908  :069,024,101,062,201,200,165
37914  :176,002,133,069,165,066,125
37920  :041,004,240,023,165,067,060
37926  :056,229,062,133,253,165,168
37932  :068,233,000,133,254,048,012
37938  :008,165,253,133,067,165,073
37944  :254,133,068,165,066,041,015
37950  :008,240,029,165,067,024,083
```

```
37956  :101,062,133,253,165,068,082
37962  :105,000,133,254,240,006,044
37968  :165,253,201,064,176,008,179
37974  :165,253,133,067,165,254,099
37980  :133,068,165,067,166,060,239
37986  :240,002,041,254,024,105,252
37992  :013,141,000,208,165,068,187
37998  :105,000,141,016,208,165,233
38004  :069,105,043,141,001,208,171
38010  :165,057,208,003,076,103,222
38016  :145,165,060,208,012,165,115
38022  :059,201,002,208,006,032,130
38028  :231,148,076,103,145,032,107
38034  :194,148,076,103,145,165,209
38040  :069,074,074,074,170,165,010
38046  :067,069,069,041,248,069,209
38052  :069,024,125,128,203,133,078
38058  :251,165,068,125,160,203,118
38064  :133,252,165,067,041,007,073
38070  :166,060,240,004,041,254,179
38076  :005,070,170,160,000,096,177
38082  :032,151,148,165,079,208,209
38088  :012,165,059,208,016,177,069
38094  :251,061,224,203,076,229,226
38100  :148,177,251,093,192,203,252
38106  :076,229,148,177,251,061,136
38112  :224,203,029,192,203,145,196
38118  :251,165,059,208,001,096,242
38124  :165,069,074,074,074,168,092
38130  :165,068,074,165,067,106,119
38136  :074,074,024,121,064,203,040
38142  :133,253,185,096,203,105,205
38148  :000,133,254,160,000,165,204
38154  :059,201,001,208,017,177,161
38160  :253,041,015,133,251,165,106
38166  :061,010,010,010,010,005,128
38172  :251,145,253,096,201,002,208
38178  :208,009,177,253,041,240,194
38184  :005,061,145,253,096,165,253
```

```
38190  :254,073,084,133,254,165,241
38196  :061,145,253,096,165,067,071
38202  :133,084,165,068,133,085,214
38208  :165,069,133,086,032,106,143
38214  :149,165,084,133,067,165,065
38220  :085,133,068,165,086,133,234
38226  :069,165,058,201,002,208,017
38232  :016,165,079,208,012,165,221
38238  :084,133,081,165,085,133,007
38244  :082,165,086,133,083,096,233
38250  :165,058,201,003,208,043,016
38256  :165,060,240,006,165,067,047
38262  :041,254,133,067,165,067,077
38268  :024,101,074,133,067,144,155
38274  :002,230,068,165,068,240,135
38280  :006,165,067,201,064,176,047
38286  :011,032,130,151,240,006,200
38292  :032,194,148,076,122,149,101
38298  :096,165,084,056,229,081,097
38304  :133,087,165,085,229,082,173
38310  :133,088,165,086,056,229,155
38316  :083,133,089,160,001,162,032
38322  :000,165,082,197,085,144,083
38328  :025,208,006,165,084,197,101
38334  :081,176,017,160,255,162,017
38340  :255,165,081,056,229,084,042
38346  :133,087,165,082,229,085,215
38352  :133,088,132,100,134,101,128
38358  :160,001,165,086,197,083,138
38364  :176,009,160,255,165,083,044
38370  :056,229,086,133,089,132,183
38376  :102,169,000,133,098,133,099
38382  :096,166,087,164,088,208,023
38388  :014,228,089,176,010,166,159
38394  :089,032,011,150,133,096,249
38400  :076,020,150,032,011,150,183
38406  :133,098,076,020,150,132,103
38412  :091,152,074,134,090,138,179
38418  :106,096,169,000,133,094,104
```

```
38424 :133,095,133,097,133,099,202
38430 :165,081,133,067,165,082,211
38436 :133,068,165,083,133,069,175
38442 :165,090,024,105,001,133,048
38448 :092,165,091,105,000,133,122
38454 :093,165,060,240,014,165,023
38460 :254,197,069,208,008,165,193
38466 :067,041,254,197,253,240,094
38472 :003,032,194,148,165,067,169
38478 :041,254,133,253,165,069,225
38484 :133,254,165,096,024,101,089
38490 :087,133,096,165,097,101,001
38496 :088,133,097,197,091,240,174
38502 :004,144,033,208,006,165,150
38508 :096,197,090,144,025,165,057
38514 :096,229,090,133,096,165,155
38520 :097,229,091,133,097,165,164
38526 :067,024,101,100,133,067,106
38532 :165,068,101,101,133,068,000
38538 :165,098,024,101,089,133,236
38544 :098,165,099,105,000,133,232
38550 :099,197,091,240,004,144,157
38556 :027,208,006,165,098,197,089
38562 :090,144,019,165,098,229,139
38568 :090,133,098,165,099,229,214
38574 :091,133,099,165,069,024,243
38580 :101,102,133,069,230,094,141
38586 :208,002,230,095,165,095,213
38592 :197,093,144,006,165,094,123
38598 :197,092,176,003,076,055,029
38604 :150,096,169,000,133,063,047
38610 :165,060,240,006,165,067,145
38616 :041,254,133,067,169,000,112
38622 :133,072,133,071,165,068,096
38628 :208,004,165,067,240,031,175
38634 :165,067,056,229,074,133,190
38640 :067,165,068,233,000,133,138
38646 :068,032,130,151,208,230,041
38652 :165,067,024,101,074,133,048
```

```
38658  :067,165,068,105,000,133,028
38664  :068,230,069,032,130,151,176
38670  :240,013,165,072,208,013,213
38676  :032,178,151,169,001,133,172
38682  :072,208,004,169,000,133,100
38688  :072,198,069,198,069,032,158
38694  :130,151,240,013,165,071,040
38700  :208,013,032,178,151,169,027
38706  :001,133,071,208,004,169,124
38712  :000,133,071,230,069,032,079
38718  :194,148,165,067,024,101,249
38724  :074,133,067,165,068,105,168
38730  :000,133,068,165,068,240,236
38736  :006,165,067,201,064,176,247
38742  :005,032,130,151,208,173,017
38748  :164,063,240,101,032,228,152
38754  :255,201,000,208,094,136,224
38760  :185,000,157,133,069,185,065
38766  :000,158,133,068,185,000,142
38772  :159,133,067,132,063,165,067
38778  :069,201,200,176,221,076,041
38784  :220,150,032,151,148,189,250
38790  :224,203,073,255,049,251,165
38796  :072,138,041,007,170,104,160
38802  :228,073,176,006,074,232,167
38808  :228,073,144,250,166,080,069
38814  :208,005,197,059,076,177,112
38820  :151,162,001,008,201,000,175
38826  :240,004,040,162,000,096,200
38832  :040,096,164,063,165,067,003
38838  :153,000,159,165,068,153,112
38844  :000,158,165,069,153,000,221
38850  :157,230,063,096,032,021,025
38856  :152,201,255,208,001,096,089
38862  :201,000,208,007,173,000,027
38868  :220,073,127,133,066,165,228
38874  :066,041,016,208,053,165,255
38880  :066,041,003,240,016,010,088
38886  :056,233,003,073,254,024,105
```

```
38892  :109,033,208,141,033,208,200
38898  :076,006,152,165,066,041,236
38904  :012,240,203,074,056,233,042
38910  :003,024,109,032,208,141,003
38916  :032,208,162,064,160,255,117
38922  :072,104,136,208,251,202,215
38928  :208,248,240,178,096,165,127
38934  :197,201,064,208,005,169,098
38940  :000,133,066,096,162,007,236
38946  :221,165,155,208,006,189,210
38952  :173,155,133,066,096,202,097
38958  :016,242,169,000,133,066,160
38964  :169,255,096,160,039,185,188
38970  :192,143,153,064,191,185,218
38976  :192,219,153,112,191,165,072
38982  :061,153,192,219,169,032,128
38988  :153,192,143,136,016,231,179
38994  :169,027,133,075,169,053,196
39000  :133,076,169,008,133,077,172
39006  :096,160,039,185,064,191,061
39012  :153,192,143,185,112,191,052
39018  :153,192,219,136,016,241,039
39024  :169,059,133,075,169,056,005
39030  :133,076,165,078,133,077,012
39036  :096,120,169,127,141,013,022
39042  :220,169,001,141,026,208,127
39048  :169,000,141,018,208,173,077
39054  :017,208,041,127,141,017,181
39060  :208,173,020,003,141,034,215
39066  :153,173,021,003,141,035,168
39072  :153,169,211,141,020,003,089
39078  :169,152,141,021,003,088,228
39084  :169,001,141,021,208,096,040
39090  :169,000,141,026,208,173,127
39096  :013,220,009,129,141,013,197
39102  :220,120,173,034,153,141,007
39108  :020,003,173,035,153,141,209
39114  :021,003,088,169,000,141,112
39120  :021,208,096,173,025,208,171
```

```
39126  :141,025,208,041,001,240,102
39132  :071,165,075,141,017,208,129
39138  :165,076,141,024,208,165,237
39144  :077,141,022,208,162,242,060
39150  :160,001,173,018,208,016,046
39156  :004,162,000,160,000,142,200
39162  :018,208,173,017,208,041,147
39168  :127,141,017,208,192,000,173
39174  :208,003,076,026,153,169,129
39180  :059,141,017,208,169,056,150
39186  :141,024,208,165,078,141,007
39192  :022,208,173,013,220,041,189
39198  :001,240,003,076,049,234,121
39204  :104,168,104,170,104,064,238
39210  :032,055,152,162,253,160,088
39216  :153,032,049,154,240,072,236
39222  :169,000,133,002,173,000,019
39228  :002,201,076,240,006,201,018
39234  :083,208,057,230,002,162,040
39240  :011,160,154,032,049,154,120
39246  :201,001,208,044,173,000,193
39252  :002,056,233,048,162,003,076
39258  :221,037,154,240,005,202,181
39264  :016,248,048,026,188,041,151
39270  :154,133,063,170,224,001,079
39276  :208,002,164,002,169,001,142
39282  :032,186,255,162,026,160,167
39288  :154,032,049,154,208,006,211
39294  :032,095,152,076,103,145,217
39300  :165,063,201,008,144,018,219
39306  :165,002,240,014,160,000,207
39312  :185,045,154,157,000,002,175
39318  :232,200,192,004,208,244,206
39324  :138,162,000,160,002,032,138
39330  :189,255,032,095,152,032,149
39336  :178,152,032,149,154,169,234
39342  :001,032,195,255,032,125,046
39348  :152,165,063,201,008,144,145
39354  :060,032,055,152,169,015,157
```

```
39360  :168,166,063,032,186,255,038
39366  :169,000,032,189,255,032,107
39372  :192,255,162,015,032,198,034
39378  :255,160,000,032,207,255,095
39384  :201,013,240,011,041,063,017
39390  :153,192,143,200,032,183,101
39396  :255,240,238,169,015,032,153
39402  :195,255,169,150,133,162,018
39408  :165,162,208,252,032,095,130
39414  :152,032,231,255,076,103,071
39420  :145,012,015,001,004,032,205
39426  :015,018,032,019,001,022,109
39432  :005,063,000,004,005,022,107
39438  :009,003,005,032,014,021,098
39444  :013,002,005,018,063,000,121
39450  :006,009,012,005,032,014,104
39456  :001,013,005,058,000,001,110
39462  :002,008,009,001,000,002,060
39468  :002,044,083,044,087,134,182
39474  :253,132,254,160,039,169,033
39480  :032,153,192,143,136,016,216
39486  :250,200,177,253,240,006,164
39492  :153,192,143,200,208,246,186
39498  :200,162,000,169,160,153,150
39504  :192,143,132,251,134,252,160
39510  :032,228,255,240,251,164,232
39516  :251,166,252,201,013,208,159
39522  :007,169,032,153,192,143,026
39528  :138,096,201,020,208,014,013
39534  :224,000,240,219,169,032,226
39540  :153,192,143,136,202,076,250
39546  :077,154,201,032,144,205,167
39552  :201,096,176,201,192,039,009
39558  :240,197,157,000,002,041,003
39564  :063,153,192,143,200,232,099
39570  :076,077,154,032,192,255,164
39576  :176,016,032,183,255,208,254
39582  :011,162,001,165,002,208,195
39588  :006,032,198,255,144,006,037
```

```
39594  :096,032,201,255,176,250,156
39600  :032,183,255,208,245,169,244
39606  :208,133,252,169,032,133,085
39612  :251,032,093,155,230,251,176
39618  :032,093,155,169,000,133,008
39624  :251,169,140,133,252,032,153
39630  :093,155,230,251,208,249,112
39636  :230,252,165,252,201,143,175
39642  :208,241,032,093,155,230,153
39648  :251,165,251,201,232,208,252
39654  :245,169,000,133,251,169,173
39660  :216,133,252,032,093,155,093
39666  :230,251,208,249,230,252,126
39672  :165,252,201,219,208,241,254
39678  :032,093,155,230,251,165,156
39684  :251,201,232,208,245,169,030
39690  :000,133,251,169,160,133,088
39696  :252,032,093,155,160,000,196
39702  :177,251,208,061,165,002,118
39708  :208,034,032,207,255,133,129
39714  :253,032,207,255,133,254,144
39720  :169,000,168,145,251,032,037
39726  :146,155,176,042,165,251,213
39732  :197,253,208,240,165,252,087
39738  :197,254,208,234,240,209,120
39744  :032,146,155,144,006,032,067
39750  :135,155,076,092,155,160,075
39756  :000,177,251,240,239,032,247
39762  :135,155,032,093,155,032,172
39768  :146,155,144,181,096,165,207
39774  :002,208,021,032,207,255,051
39780  :072,176,028,032,183,255,078
39786  :240,004,201,064,208,019,074
39792  :104,160,000,145,251,096,100
39798  :160,000,177,251,032,210,180
39804  :255,032,183,255,208,002,035
39810  :096,104,104,104,096,165,031
39816  :251,032,210,255,165,252,021
39822  :032,210,255,096,230,251,192
```

```
39828 :208,004,230,252,024,096,194
39834 :165,252,201,191,144,004,087
39840 :165,251,201,065,096,018,188
39846 :023,010,009,020,012,062,046
39852 :014,008,002,004,001,010,211
39858 :006,005,009,144,005,028,119
39864 :159,156,030,031,158,129,079
39870 :149,150,151,152,153,154,075
39876 :155,136,135,134,133,072,193
39882 :073,082,069,083,032,083,112
39888 :075,069,084,067,072,080,143
39894 :065,068,032,045,032,066,010
39900 :089,032,067,072,082,073,123
39906 :083,032,077,069,084,067,126
39912 :065,076,070,013,077,085,106
39918 :076,084,073,067,079,076,181
39924 :079,082,032,077,079,068,149
39930 :069,063,032,078,157,000,137
39936 :016,012,015,020,058,006,127
39942 :032,058,032,032,032,032,224
39948 :032,012,009,014,005,032,116
39954 :004,018,001,023,058,032,154
39960 :032,032,032,032,032,006,190
39966 :009,012,012,020,015,058,156
39972 :032,032,032,032,055,053,016
39978 :051,049,015,006,006,032,201
39984 :015,014,032,032,015,006,162
39990 :006,032,006,018,015,013,144
39996 :020,015,032,032,012,009,180
40002 :014,005,019,001,013,005,123
40008 :001,014,025,032,000,056,200
40014 :000,000,068,000,000,068,214
40020 :000,006,254,192,009,001,034
40026 :032,006,000,192,004,000,068
40032 :064,004,000,064,004,000,232
40038 :064,006,000,192,009,001,118
40044 :032,006,254,192,000,056,136
40050 :000,000,012,000,024,000,150
40056 :000,036,000,000,036,000,192
```

```
40062 :003,126,192,004,129,032,100
40068 :003,000,192,002,000,064,137
40074 :002,000,064,002,000,064,014
40080 :003,000,192,004,129,032,248
40086 :003,126,192,000,024,000,239
40092 :000,012,000,000,255,255,166
```

Programma 2. Sottoprogramma di caricamento e registrazione

```
100 REM SEGNALATE A QUESTO SOTTOPROGRAM-
110 REM MA SE VOLETE REGISTRARE O LEGGE-
120 REM RE TRAMITE LA VARIABILE LS ( 0=
130 REM LOAD, 1= SAVE). SEGNALATE IL NO-
140 REM ME DEL FILE TRAMITE NM$ (CON 0:
150 REM O @0: SE DESIDERATE) E LA PERI-
160 REM FERICA CON DV. QUINDI INSERITE
163 REM UNA ISTRUZIONE GOSUB 50000.
165 REM IL SOTTOPROGRAMMA SI INTERROMPE
167 REM IN CASO DI ERRORE, E POTETE LEG-
169 REM GERE IL CANALE OPPORTUNO.
170 :
1000 I=51676
1010 READA:IFA>=0THENPOKEI,A:I=I+1:GOTO101
     0
1020 REM SEMPLICE PROGRAMMA ESEMPLIFICATIV
     O
1030 INPUT"LOAD O SAVE";A$:LS=0:IFLEFT$(A$
     ,1)="S"THENLS=1
1040 INPUT"PERIFERICA";DV
1050 INPUT"NOME";NM$
1060 GOSUB50000:END
2000 :
50000 ZS=2:IF(LS<>1ANDLS<>0)ORNM$=""ORLEN(
      NM$)>16ORDV<1ORDV>11THENRETURN
```

```
50010 IFDV<3THENZS=0:IFDV<2THENZS=LS
50020 IFDV>7ANDLS=1THENNM$=NM$+",S,W"
50030 IFLS=1THENPRINT"{GIU'}SAVING{SPAZI}"
      NM$:POKE2,1:GOTO50050
50040 PRINT"{GIU'}LOADING "NM$:POKE2,0:POK
      E58576,197:POKE53272,56:POKE53265,59

50050 OPEN1,DV,ZS,NM$:SYS(51676):POKE56576
      ,199:POKE53272,21:POKE52165,27
50060 RETURN
50070 :
51676 DATA169,0,133,157,32,183,255,208,7,1
      69,54,133,1,32,249,201
51692 DATA169,1,32,195,255,32,231,255,169,
      55,133,1,96,162,1,165
51708 DATA2,208,6,32,198,255,144,6,96,32,2
      01,255,176,250,32,183
51724 DATA255,208,245,169,208,133,252,169,
      32,133,251,32,183,202,230,251
51740 DATA32,183,202,169,0,133,251,169,140
      ,133,252,32,183,202,230,251
51756 DATA208,249,230,252,165,252,201,143,
      208,241,32,183,202,230,251,165
51772 DATA251,201,232,208,245,169,0,133,25
      1,169,216,133,252,32,183,202
51788 DATA230,251,208,249,230,252,165,252,
      201,219,208,241,32,183,202,230
51804 DATA251,165,251,201,232,208,245,169,
      0,133,251,169,160,133,252,32
51820 DATA183,202,160,0,177,251,208,61,165
      ,2,208,34,32,228,255,133
51836 DATA253,32,228,255,133,254,169,0,168
      ,145,251,32,236,202,176,42
51852 DATA165,251,197,253,208,240,165,252,
      197,254,208,234,240,209,32,236
51868 DATA202,144,6,32,225,202,76,182,202,
      160,0,177,251,240,239,32
51884 DATA225,202,32,183,202,32,236,202,14
      4,181,96,165,2,208,21,32
```

```
51900 DATA228,255,72,176,28,32,183,255,240
      ,4,201,64,208,19,104,160
51916 DATA0,145,251,96,160,0,177,251,32,21
      0,255,32,183,255,208,2
51932 DATA96,104,104,104,96,165,251,32,210
      ,255,165,252,32,210,255,96
51948 DATA230,251,208,4,230,252,24,96,165,
      252,201,191,144,4,165,251
51964 DATA201,65,96,255,-1
```

Programma 3. Codice sorgente del blocco per schizzi

```
                ;BREAKDOWN OF MEMORY USAGE
9000
                ; $0400-$07E7 TEXT SCREEN (BASIC)
                ; $0801-$8BFF BASIC PROGRAM AREA (UNUSED)
                ; $8C00-$8FE7 TEMPORARY COLOR TEXT SCREEN
                ; $9000-$9C9F 3232 BYTES FOR MAIN PROGRAM
                ; $9D00-$9FFF FILL STACK STORAGE
                ; $A000-$BF3F HI-RES MAP PAGE
                ; $BF40-$BF97 ADDITIONAL STORAGE
                ; $BFC0-$BFFF SPRITE 0 BLOCK
                ; $CB00-$CBFF ADDITIONAL STORAGE
                ; $D800-$DBE7 COLOR SCREEN
                ;
                ;LIST OF PROGRAM VARIABLES
                ;
9000           DRAW    =   57     ;WHETHER TO PLOT WHILE MOVING
9000           LINES   =   58     ;TYPE OF LINES TO DRAW
9000           PLOTMD  =   59     ;TYPE OF PLOTTING
9000           SCRNMD  =   60     ;0 = HIRES, 1 = MULTICOLOR
9000           PLCOL   =   61     ;PLOTTING COLOR
9000           PLINC   =   62     ;PLOT INCREMENT
9000           PNTR    =   63     ;FILL POINTER/DEVICE NUMBER
9000           PRESSD  =   64     ;FIRE BUTTON PRESSED
9000           TIMSTO  =   65     ;LAST JIFFY VALUE
9000           JOY     =   66     ;CURRENT JOYSTICK VALUE
9000           XPOS    =   67     ;(2) X-POSITION OF PLOTTER
9000           YPOS    =   69     ;Y-POSITION OF PLOTTER
9000           AD      =   251    ;(2) WHERE BIT IS PLOTTED
9000           TYPE    =   70     ;TYPE OF PLOT
9000           MISC    =   253    ;(2) MISC COUNTER/POINTER
9000           UPF     =   71     ;FILL FLAG FOR UP DIRECTION
9000           DOWNF   =   72     ;FILL FLAG FOR DOWN
9000           FLSHFT  =   73     ;HOW FAR TO SHIFT A BYTE (7/6)
9000           FLLINC  =   74     ;INCREMENT FOR FILL (1/2)
9000           HIR1    =   75     ;HIRES VIC CHIP SHADOWS (SCREEN BANK)
9000           HIR2    =   76     ; (HIRES/TEXT)
9000           HIR3    =   77     ; (MULTICOLOR)
9000           HIR4    =   78     ; (DISPLAY MULTICOLOR ON/OFF)
9000           XORIT   =   79     ;FLAG FOR RUBBERBAND
9000           FILBOR  =   80     ;FILL TO WHAT
```

```
9000    LDSV     =   2          ;LOAD/SAVE FLAG
        ;
9000    X1       =   81         ;(2) X START
9000    Y1       =   83         ;Y START
9000    X2       =   84         ;(2) X END
9000    Y2       =   86         ;Y END
9000    XDIFF    =   87         ;(2) ABSOLUTE X DIFFERENCE
9000    YDIFF    =   89         ;Y DIFFERENCE
9000    DIS      =   90         ;(2) GREATER DISTANCE
9000    DIS2     =   92         ;(2) HOLDS DIS+2
9000    CNTR     =   94         ;(2) COUNTER FOR DISTANCE
9000    XCNTR    =   96         ;(2) COUNTER FOR NEXT PLOT
9000    YCNTR    =   98         ;(2) NEXT PLOT COUNTER
9000    XSGN     =   100        ;(2) SIGNUM OF X-DIFFERENCE
9000    YSGN     =   102        ;YDIFF SGN

        ;
9000    TABLE1   =   $9D00      ;FILL STACKS
9000    TABLE2   =   $9E00
9000    TABLE3   =   $9F00
9000    SCLINE   =   $BF40      ;COPY OF BOTTOM SCREEN LINE
9000    COLINE   =   $BF70      ;BOTTOM COLOR SCREEN LINE
9000    ZERSTO   =   $CB00      ;57-102 ZERO PAGE STORAGE
9000    FORTYL   =   $CB40      ;TABLE OF ADDRESSES
9000    FORTYH   =   $CB60      ;  OF SCREEN LINES
9000    LINEL    =   $CB80      ;TABLE OF ADDRESSES
9000    LINEH    =   $CBA0      ;  OF LINES ON HIRES PAGE
9000    PIXEL    =   $CBC0      ;HIRES/MULTI PIXELS
9000    MASK     =   $CBE0      ;PIXEL MASKS

        ;
        ;LIST OF SYSTEM LOCATIONS
        ;
9000    R6510    =   $01        ;ON-CHIP MEMERY CONTROL REGISTER
9000    MEMSIZ   =   $37        ;BASIC TOP OF MEMORY
9000    MSGFLG   =   $9D        ;KERNAL MESSAGES ON/OFF
9000    TIME     =   $A0        ;3-BYTE TIMER (HML)
9000    LSTX     =   $C5        ;CURRENT KEY PRESSED (64=NONE)
9000    INBUFF   =   $0200      ;LINE INPUT BUFFER
9000    COLOR    =   $0286      ;CURRENT CHARACTER COLOR
9000    RPTFLG   =   $028A      ;KEY REPEAT ON/OFF
9000    KOUNT    =   $028B      ;KEY REPEAT SPEED
9000    SHFLAG   =   $028D      ;SHIFT/CTRL/C= FLAG
9000    INTPNT   =   $0314      ;IRQ POINTER
9000    CRT      =   $8C00      ;TEMPORARY TEXT SCREEN
9000    S0PNTR   =   CRT+1016     ;SPRITE 0 POINTER
9000    HIPAGE   =   $A000      ;START OF HIRES PAGE
9000    SBLOCK   =   $BFC0      ;SPRITE BLOCK ADDRESS
9000    VIC      =   $D000      ;START OF VIC CHIP
9000    JSTICK   =   $DC00      ;JOYSTICK #2 STATUS
9000    CIAICR   =   $DC0D      ;CIA INTERRUPT CONTROL REG.
9000    COLCRT   =   $D800      ;START OF COLOR SCREEN
9000    VBANK    =   $DD00      ;16K BANK SELECTER FOR VIC

        ;
9000    INTRPT   =   $EA31      ;NORMAL IRQ VECTOR
9000    CHKIN    =   $FFC6      ;OPEN INPUT CHANNEL
9000    CHKOUT   =   $FFC9      ;OPEN OUTPUT CHANNEL
9000    CHRIN    =   $FFCF      ;INPUT A BYTE
9000    CHROUT   =   $FFD2      ;PRINT CHR$(.A)
9000    CLALL    =   $FFE7      ;CLOSE ALL
9000    CLOSE    =   $FFC3      ;CLOSE A FILE
9000    GETIN    =   $FFE4      ;GET CHARACTER
9000    OPEN     =   $FFC0      ;OPEN THE FILE
9000    READST   =   $FFB7      ;CHECK STATUS WORD
9000    SETLFS   =   $FFBA      ;SET FILE PARAMETERS
9000    SETNAM   =   $FFBD      ;SET FILE NAME
        ;
        ;INITIALIZE USER INFORMATION
        ;
```

Capitolo due

```
9000 20 E7 FF PROGRM    JSR   CLALL     ;CLOSE ALL OPEN FILES
9003 A0 00              LDY   #0        ;DISPLAY "MULTICOLOR MODE? N"
9005 B9 C9 9B QLOOP     LDA   QUESTN,Y   ;GET A CHARACTER
9008 F0 06              BEQ   QEND      ;ZERO BYTE FLAGS THE END
900A 20 D2 FF           JSR   CHROUT    ;PRINT THE CHARACTER

900D C8                 INY             ;NEXT BYTE
900E D0 F5              BNE   QLOOP     ;BRANCH-ALWAYS
9010 A0 00     QEND     LDY   #0        ;PREPARE FOR A 'NO'
9012 20 CF FF           JSR   CHRIN     ;GET OUR INPUT
9015 C9 0D              CMP   #13       ;JUST A RETURN
9017 D0 01              BNE   NYCK      ;NO
9019 60                 RTS             ;YES, SO ABORT
901A C9 59     NYCK     CMP   #"Y"      ;CHECK IF "YES"
901C D0 01              BNE   HIPICK    ;NO
901E C8                 INY             ;SET .Y TO 1
901F 98        HIPICK   TYA
9020 48                 PHA             ;SAVE .Y ON STACK
9021 20 CF FF CLRIN     JSR   CHRIN     ;PULL THE REST OF THE INPUT
9024 C9 0D              CMP   #13
9026 D0 F9              BNE   CLRIN     ;UNTIL A RETURN COMES UP
               ;
9028 A0 2D              LDY   #45
902A B9 39 00 ZEROFF    LDA   57,Y      ;STORE 57-102 ZERO PAGE
902D 99 00 CB           STA   ZERSTO,Y   ;... AT $CB00
9030 88                 DEY
9031 10 F7              BPL   ZEROFF    ;DO 45 TO 0
9033 68                 PLA
9034 85 3C              STA   SCRNMD    ;PUT OLD .Y IN SCRNMD


               ;
               ;SET UP SYSTEM FOR HIGH RESOLUTION
               ;
9036 A9 00     SETUP    LDA   #0        ;SET PROGRAM MODE
9038 85 9D              STA   MSGFLG    ;$00=PROGRAM,$80=IMMEDIATE
903A A9 8C              LDA   #>CRT     ;SET TOP OF MEMORY AND
903C 85 38              STA   MEMSIZ+1   ; BASIC STRING POINTER
903E 85 34              STA   MEMSIZ-3   ; TO START OF TEMPORARY
9040 A9 00              LDA   #<CRT     ; SCREEN MEMORY
9042 85 37              STA   MEMSIZ
9044 85 33              STA   MEMSIZ-4
9046 A9 80              LDA   #$80      ;SET ALL-KEY REPEAT MODE
9048 8D 8A 02           STA   RPTFLG    ;$0=CURSOR,$40=NONE,$80=ALL REPEAT
904B A9 C5              LDA   #196+1    ;1 INDICATES BANK 2
904D 8D 00 DD           STA   VBANK     ;VIC SEES $8000
9050 A9 36              LDA   #55-1     ;FLIP OUT BASIC
9052 85 01              STA   R6510
9054 A9 38              LDA   #8+48     ;HIRES $A000, TEXT $8C00
9056 85 4C              STA   HIR2      ;VIC+24 SHADOW
9058 A9 3B              LDA   #27+32    ;32 SETS HIRES GRAPHICS
905A 85 4B              STA   HIR1      ;VIC+17 SHADOW
905C A9 08              LDA   #8        ;ASSUME NO MULTICOLOR ...
905E A4 3C              LDY   SCRNMD    ;CHECK IT
9060 F0 02              BEQ   SETHIR    ;NONE
9062 A9 18              LDA   #8+16     ;16 SETS MULTICOLOR
9064 85 4D     SETHIR   STA   HIR3
9066 85 4E              STA   HIR4      ;VIC+22 SHADOWS


               ;
               ;INITIALIZE PROGRAM VARIABLES
               ;
9068 A9 00     SETVAR   LDA   #0
906A 85 40              STA   PRESSD    ;JOYSTICK NOT PRESSED
906C 85 39              STA   DRAW      ;BEGIN WITH PEN UP
906E 85 3A              STA   LINES     ;LINE DRAW OFF
9070 85 4F              STA   XORIT     ;NO RUBBERBAND LINE
9072 85 50              STA   FILBOR    ;FILL TO SAME PIXEL PATTERN
9074 85 44              STA   XPOS+1    ;SET XPOS TO 160
9076 A9 A0              LDA   #160
```

```
9078 85 43              STA   XPOS
907A A9 64              LDA   #100      ;SET YPOS TO 100
907C 85 45              STA   YPOS
907E A9 01              LDA   #1
9080 85 3E              STA   PLINC     ;INITIAL PLOT INCREMENT 1
9082 85 3B              STA   PLOTMD    ;F5 PLOT MODE
9084 A5 3C              LDA   SCRNMD
9086 0A                 ASL   A         ;MULTIPLY SCRNMD BY 8
9087 0A                 ASL   A
9088 0A                 ASL   A
9089 85 46              STA   TYPE      ;SET PIXEL TABLE POINTER
908B A9 07              LDA   #7        ;SET FILL SHIFT BY SCRNMD (7 OR 6)
908D 38                 SEC
908E E5 3C              SBC   SCRNMD
9090 85 49              STA   FLSHFT
9092 A9 01              LDA   #1        ;SET FILL INCREMENT (0 OR 1)
9094 18                 CLC
9095 65 3C              ADC   SCRNMD
9097 85 4A              STA   FLLINC

                ;
                ;SET UP TABLES (ADDRESSES,PIXELS,MASKS)
                ;
9099 A9 8C              LDA   #>CRT     ;SET COLOR SCREEN ADDRESSES FROM $8C00
909B 8D 60 CB           STA   FORTYH
909E A9 00              LDA   #<CRT
90A0 8D 40 CB           STA   FORTYL
90A3 A9 A0              LDA   #>HIPAGE  ;SET HIRES LINE ADDRESSES FROM $A000
90A5 8D A0 CB           STA   LINEH
90A8 A9 00              LDA   #<HIPAGE
90AA 8D 80 CB           STA   LINEL
90AD AA                 TAX             ;<HIPAGE = 0 THEREFORE LDX #0
90AE BD 40 CB FLOOP     LDA   FORTYL,X  ;GET PREVIOUS ENTRY
90B1 18                 CLC
90B2 69 28              ADC   #40       ;ADD 40
90B4 9D 41 CB           STA   FORTYL+1,X  ;STORE AT NEXT ENTRY
90B7 BD 60 CB           LDA   FORTYH,X  ;GET HIGH BYTE (DIFFERENT TABLE)
90BA 69 00              ADC   #0        ;INCREMENT IF NECESSARY
90BC 9D 61 CB           STA   FORTYH+1,X
90BF BD 80 CB           LDA   LINEL,X   ;GET PREVIOUS
90C2 18                 CLC
90C3 69 40              ADC   #64       ;ADD 64 TO LOW BYTE
90C5 9D 81 CB           STA   LINEL+1,X
90C8 BD A0 CB           LDA   LINEH,X
90CB 69 01              ADC   #1        ;AND 1 OR 2 TO HIGH BYTE
90CD 9D A1 CB           STA   LINEH+1,X
90D0 E8                 INX             ;NEXT ROW
90D1 E0 18              CPX   #24       ;ASSIGNED ALL 25 ROWS
90D3 D0 D9              BNE   FLOOP

                ;
90D5 A9 01              LDA   #%00000001
90D7 A0 07              LDY   #7        ;SET UP FOR HIRES PIXELS
90D9 99 C0 CB PXLP1     STA   PIXEL,Y
90DC 0A                 ASL   A         ;SHIFT IT
90DD 88                 DEY             ; NEXT ENTRY
90DE 10 F9              BPL   PXLP1
90E0 A9 01              LDA   #%00000001
90E2 A0 06              LDY   #6        ;SET UP FOR F3/F5 MULTICOLOR
90E4 99 C8 CB PXLP2     STA   PIXEL+8,Y
90E7 0A                 ASL   A         ;SHIFT FOR F3
90E8 99 D0 CB           STA   PIXEL+16,Y
90EB 0A                 ASL   A         ;SHIFT FOR F5
90EC 88                 DEY             ;SKIP AN ENTRY
90ED 88                 DEY
90EE 10 F4              BPL   PXLP2
90F0 A9 03              LDA   #%00000011
90F2 A0 06              LDY   #6        ;SET UP FOR MULTICOLOR F1
90F4 99 D8 CB PXLP3     STA   PIXEL+24,Y
```

```
90F7 ØA                    ASL  A          ;SHIFT TWICE
90F8 ØA                    ASL  A
90F9 88                    DEY             ;SKIP AN ENTRY
90FA 88                    DEY
90FB 10 F7                 BPL  PXLP3

90FD A9 FE                 LDA  #$FE
90FF A0 07                 LDY  #7         ;SET UP MASK TABLES
9101 99 E0 CB MASK1        STA  MASK,Y
9104 38                    SEC
9105 2A                    ROL  A          ;ROLL ON A BIT
9106 88                    DEY
9107 10 F8                 BPL  MASK1      ;DO 7 THROUGH Ø
9109 A9 FC                 LDA  #%11111100
910B A0 07                 LDY  #7         ;SET UP FOR MULTI MASKS
910D 99 E7 CB MASK2        STA  MASK+7,Y

9110 99 EF CB              STA  MASK+15,Y
9113 99 F7 CB              STA  MASK+23,Y
9116 38                    SEC             ;ROLL ON TWO BITS
9117 2A                    ROL  A
9118 38                    SEC
9119 2A                    ROL  A
911A 88                    DEY
911B 88                    DEY
911C 10 EF                 BPL  MASK2

                  ;
                  ;ASSIGN SPRITE VARIABLES
                  ;
911E A0 28                 LDY  #40        ;COUNTER FOR BYTES LOADED
9120 A2 28                 LDX  #40        ;TOP OF HIRES SPRITE
9122 A5 3C                 LDA  SCRNMD
9124 F0 02                 BEQ  SLOOP      ;HIRES
9126 A2 51                 LDX  #81        ;TOP OF MULTICOLOR SPRITE
9128 BD 4C 9C SLOOP        LDA  SPRITE,X   ;FROM APPROPRIATE SPRITE
912B 99 C0 BF              STA  SBLOCK,Y   ;STORE IN TABLE
912E CA                    DEX             ;NEXT LOOP
912F 88                    DEY             ;COUNT ONE BYTE
9130 10 F6                 BPL  SLOOP
9132 A9 00                 LDA  #0         ;CLEAR REMAINING BYTES
9134 A0 15                 LDY  #21        ;22 BYTES REMAIN
9136 99 E9 BF LOOP0        STA  SBLOCK+41,Y  ;CLEAR THEM
9139 88                    DEY
913A 10 FA                 BPL  LOOP0
913C A9 FF                 LDA  #255       ;SPRITE BLOCK (AT 49088)
913E 8D F8 8F              STA  S0PNTR     ;SPRITE 0 POINTER
9141 A9 AC                 LDA  #172       ;160 + X-OFFSET (12)
9143 18                    CLC
9144 65 3C                 ADC  SCRNMD     ;MULTICOLOR OFFSET
9146 8D 00 D0              STA  VIC        ;HARDWARE SPRITE 0 X-REGISTER
9149 A9 8F                 LDA  #143       ;100 + Y-OFFSET (43)
914B 8D 01 D0              STA  VIC+1      ;SPRITE 0 Y-REGISTER
914E A9 00                 LDA  #0         ;TURN OFF ...
9150 8D 1B D0              STA  VIC+27     ;... BACKGROUND PRIORITY
9153 8D 1C D0              STA  VIC+28     ;... SPRITE MULTICOLOR
9156 8D 1D D0              STA  VIC+29     ;... SPRITE EXPAND X
9159 8D 17 D0              STA  VIC+23     ;... SPRITE EXPAND Y
915C AD 86 02              LDA  COLOR      ;CHARACTER COLOR
915F 85 3D                 STA  PLCOL      ;PLOTTING COLOR SHADOW
9161 8D 27 D0              STA  VIC+39     ;SPRITE 0 COLOR
9164 20 7D 98              JSR  RSTON      ;TURN ON RASTER INTERRUPT

                  ;
                  ;BEGINNING OF MAIN LOOP
                  ;
9167 A9 01    BEGIN        LDA  #1
9169 8D 8B 02              STA  KOUNT      ;SET MINIMUM REPEAT
916C 20 E4 FF              JSR  GETIN      ;PULL CHARACTER FROM KEYBOARD
```

```
916F 48             PHA          ;SAVE ON STACK
9170 A5 3A          LDA  LINES   ;CHECK FOR LINE MODE
9172 F0 28          BEQ  ANALZE  ;NO
9174 AD 8D 02       LDA  SHFLAG  ;SHIFT/CTRL/C= FLAG
9177 29 01          AND  #1      ;TEST SHIFT KEY ONLY
9179 D0 07          BNE  YESLIN  ;PUSHED DOWN
917B AD 00 DC       LDA  JSTICK  ;GET PORT 2 INPUT
917E 29 10          AND  #16     ;CHECK FIRE BUTTON
9180 D0 06          BNE  XORLIN  ;NOT PRESSED
9182 20 38 95 YESLIN JSR LINE    ;PLOT A LINE
9185 4C 9C 91       JMP  ANALZE  ;SKIP RUBBERBAND
9188 A5 3A    XORLIN LDA LINES   ;LINE MODE FLAG
918A C9 03          CMP  #3      ;CHECK FOR RIGHT FILL MODE
918C F0 0E          BEQ  ANALZE  ;IF YES, SKIP
918E A9 01          LDA  #1      ;SET RUBBERBAND FLAG
9190 85 4F          STA  XORIT
9192 20 38 95       JSR  LINE    ;DRAW IT ON
9195 20 38 95       JSR  LINE    ;PLOT IT OFF
9198 A9 00          LDA  #0      ;UNSET RUBBERBAND MODE
919A 85 4F          STA  XORIT
919C 68       ANALZE PLA         ;RESTORE GETIN CHARACTER
919D D0 03          BNE  DOKEYS  ;IF A KEY PRESSED
919F 4C BC 93       JMP  JOYSTK  ;ELSE SKIP KEY CHECKS
91A2 48       DOKEYS PHA         ;RESAVE .A ON STACK
91A3 20 15 98       JSR  KEYS    ;SET JOY IF A KEYBOARD DIRECTION
91A6 68             PLA          ;RESTORE CHARACTER
91A7 A4 42          LDY  JOY     ;TEST FOR A DIRECTION
91A9 F0 03          BEQ  SPACE   ;IF NOT, CHECK OTHERS
91AB 4C D9 93       JMP  FIRECK  ;YES, SO EXECUTE

              ;
              ;CHECK THE FUNCTIONS
              ;
91AE C9 20    SPACE  CMP  #32     ;SPACE
91B0 F0 04           BEQ  SPACDO  ;YES
91B2 C9 A0           CMP  #160    ;SHIFT SPACE
91B4 D0 09           BNE  HOME    ;IF NOT, TRY ANOTHER
91B6 A5 39    SPACDO LDA  DRAW    ;MOVE/PLOT FLAG
91B8 49 01           EOR  #1      ;TOGGLE ON/OFF
91BA 85 39           STA  DRAW    ;STORE IT BACK
91BC 4C BC 93        JMP  JOYSTK  ;GO AND CHECK JOYSTICK
91BF C9 53    HOME   CMP  #"S"    ;HOME THE PLOTTER
91C1 D0 0F           BNE  CORNER
91C3 A9 00           LDA  #0      ;CENTER THE PLOTTER
91C5 85 44           STA  XPOS+1
91C7 A9 A0           LDA  #160    ;X-CENTER
91C9 85 43           STA  XPOS
91CB A9 64           LDA  #100    ;Y-CENTER
91CD 85 45           STA  YPOS
91CF 4C BC 93        JMP  JOYSTK  ;GO TO JOYSTICK
91D2 C9 13    CORNER CMP  #19     ;CONTROL-S (HOME)
91D4 D0 0B           BNE  PLUS
91D6 A9 00           LDA  #0      ;ZERO X AND Y COORDS
91D8 85 43           STA  XPOS
91DA 85 44           STA  XPOS+1
91DC 85 45           STA  YPOS
91DE 4C BC 93        JMP  JOYSTK
91E1 C9 2B    PLUS   CMP  #"+"    ;PLOT ON (F1)
91E3 F0 04           BEQ  PLUSDO
91E5 C9 DB           CMP  #219    ;CHECK SHIFT-PLUS
91E7 D0 0B           BNE  MINUS
91E9 A9 01    PLUSDO LDA  #1      ;F5 = 1
91EB 85 3B    SET    STA  PLOTMD  ;STORE .A
91ED A9 08           LDA  #8      ;SET PIXEL/MASK POINTER
91EF 85 46           STA  TYPE
91F1 4C BC 93        JMP  JOYSTK
91F4 C9 2D    MINUS  CMP  #"-"    ;PLOT OFF (F7)
```

```
91F6 F0 04              BEQ  MINDO
91F8 C9 DD              CMP  #221       ;SHIFT-MINUS
91FA D0 05              BNE  FILLCK
91FC A9 00     MINDO    LDA  #0         ;F7 =-0
91FE 4C EB 91           JMP  SET        ;SET TYPE ETC.
9201 C9 8C     FILLCK   CMP  #140       ;F8
9203 D0 06              BNE  LINE1
9205 20 CE 96           JSR  FILL       ;EXECUTE A FILL
9208 4C BC 93           JMP  JOYSTK
920B C9 89     LINE1    CMP  #137       ;F2 = LINE DRAW FROM
920D D0 17              BNE  LINE2
920F A5 3A              LDA  LINES      ;0=OFF,1=FROM,2=TO,3=RIGHT
9211 29 01              AND  #1         ;MASK OFF ALL BUT FROM
9213 49 01              EOR  #1         ;TOGGLE THAT
9215 85 3A              STA  LINES      ;AND RESAVE
9217 A5 43     PTINIT   LDA  XPOS
9219 85 51              STA  X1         ;SET UP INITIAL POINTS
921B A5 44              LDA  XPOS+1
921D 85 52              STA  X1+1
921F A5 45              LDA  YPOS
9221 85 53              STA  Y1
9223 4C 5E 94           JMP  PLOT       ;SKIP OVER JOYSTICK
9226 C9 8A     LINE2    CMP  #138       ;F4 = DRAWTO
9228 D0 0B              BNE  LINE3
922A A5 3A              LDA  LINES      ;GET LINE MODE
922C 29 02              AND  #2         ;MASK OFF ALL BUT DRAWTO
922E 49 02              EOR  #2         ;TOGGLE
9230 85 3A              STA  LINES      ;RESAVE
9232 4C 17 92           JMP  PTINIT     ;GO BACK AND SET THE INITIAL POINT
9235 C9 8B     LINE3    CMP  #139       ;F6 = RIGHT DRAW
9237 D0 12              BNE  ASTCK
9239 A5 3A              LDA  LINES
923B D0 07              BNE  OFF        ;IF ANYTHING IS ON
923D A9 03              LDA  #3         ;TURN ON RIGHT DRAW
923F 85 3A              STA  LINES
9241 4C BC 93           JMP  JOYSTK
9244 A9 00     OFF      LDA  #0         ;TURN OFF RIGHT DRAW
9246 85 3A              STA  LINES
9248 4C BC 93           JMP  JOYSTK
924B C9 2A     ASTCK    CMP  #"*"       ;FILL TO ANY/SAME
924D D0 09              BNE  ARRCK
924F A5 50              LDA  FILBOR     ;GET SAME/ANY FILL MODE FLAG
9251 49 01              EOR  #1         ;TOGGLE IT
9253 85 50              STA  FILBOR     ;STORE
9255 4C BC 93           JMP  JOYSTK
9258 C9 5E     ARRCK    CMP  #"^"       ;UP-ARROW (BORDER/BACKGROUND)
925A D0 0A              BNE  RETCK
925C 24 C5     WAITCH   BIT  LSTX       ;OVFLOW SET BY BIT 6
925E 50 FC              BVC  WAITCH     ;WAIT UNTIL BIT 6 SET (NO KEY PRESSED)
9260 20 C6 97           JSR  BORBAK     ;ADJUST BORDER/BACKGROUND COLORS
9263 4C 67 91           JMP  BEGIN
               ;
9266 C9 0D     RETCK    CMP  #13
9268 D0 5B              BNE  CLEAR
926A 20 37 98           JSR  OUT        ;ENABLE THE STATUS LINE
926D A0 27              LDY  #39
926F B9 00 9C  MESSGE   LDA  STLINE,Y   ;GET A BYTE OF STATUS LINE
9272 99 C0 8F           STA  CRT+960,Y  ;PUT IT ON BOTTOM LINE
9275 88                 DEY
9276 10 F7              BPL  MESSGE     ;DISPLAY 39 TO 0
9278 A4 3B              LDY  PLOTMD     ;CHECK PLOT TYPE
927A B9 28 9C           LDA  FNUM,Y     ;LOAD (F)7,5,3,1
927D 8D C6 8F           STA  CRT+966    ;DISPLAY IT
9280 A5 39              LDA  DRAW       ;GET 0 OR 1 FOR MOVE/PLOT
9282 0A                 ASL  A
9283 0A                 ASL  A          ;SHIFT IT TWICE (* 4)
9284 A8                 TAY             ;TRANSFER TO Y FOR INDIRECT ACCESS
```

```
9285 A2 00                LDX  #0        ;ZERO X FOR THE DISPLAY LOOP
9287 B9 2C 9C ONOFF1      LDA  DRMD,Y    ;GET AN "ON  " OR "OFF " BYTE
928A 9D C8 8F             STA  CRT+968,X  ;DISPLAY IT
928D C8                   INY            ;NEXT BYTE OF MESSAGE
7928E E8                  INX            ;NEXT SCREEN BYTE
928F E0 04                CPX  #4        ;LAST BYTE DISPLAYED
9291 D0 F4                BNE  ONOFF1    ;NO
9293 A5 3A                LDA  LINES     ;GET LINE DRAW MODE (0-3)
9295 0A                   ASL  A
9296 0A                   ASL  A         ;MULTIPLY BY 4
9297 A8                   TAY
9298 A2 00                LDX  #0        ;SET UP INDEXERS
929A B9 34 9C ONOFF2      LDA  LNMD,Y    ;GET A BYTE
929D 9D D7 8F             STA  CRT+983,X  ;DISPLAY
92A0 C8                   INY            ;CONTINUE TO NEXT BYTE
92A1 E8                   INX
92A2 E0 04                CPX  #4
92A4 D0 F4                BNE  ONOFF2
92A6 A5 50                LDA  FILBOR    ;SET UP FILL TYPE (ANY/SAME)
92A8 0A                   ASL  A
92A9 0A                   ASL  A
92AA A8                   TAY
92AB A2 00                LDX  #0
92AD B9 44 9C ONOFF3      LDA  FLMD,Y
92B0 9D E4 8F             STA  CRT+996,X   ;DISPLAY
92B3 C8                   INY
92B4 E8                   INX
92B5 E0 04                CPX  #4
92B7 D0 F4                BNE  ONOFF3
92B9 A5 C5    RETWT       LDA  LSTX      ;GET KEY PRESSED
92BB C9 01                CMP  #1        ;RETURN KEY YIELDS 1
92BD F0 FA                BEQ  RETWT     ;WAIT UNTIL RELEASED
92BF 20 5F 98             JSR  IN        ;BRING BACK ALL-HIRES
92C2 4C BC 93             JMP  JOYSTK
              ;
92C5 C9 93    CLEAR       CMP  #147      ;CLR KEY
92C7 D0 48                BNE  RVSCK
92C9 A0 00                LDY  #0        ;ZERO LO-BYTE ADDRESS
92CB 84 FD                STY  MISC
92CD 98                   TYA            ;ZERO .A
92CE A2 A0                LDX  #>HIPAGE   ;X USED AS MISC+1
92D0 86 FE    LOOP1       STX  MISC+1
92D2 91 FD    LOOP2       STA  (MISC),Y   ;.A = 0
92D4 C8                   INY            ;LOW BYTE
92D5 D0 FB                BNE  LOOP2     ;WAIT ONE PAGE
92D7 E8                   INX            ;INCREMENT PAGE COUNT
92D8 E0 BF                CPX  #>31*256+HIPAGE  ;LAST PAGE
92DA D0 F4                BNE  LOOP1     ;NOT YET
92DC 86 FE                STX  MISC+1    ;SAVE IT FOR LAST PAGE
92DE 91 FD    LOOP3       STA  (MISC),Y   ;CLEAR 64 BYTES
92E0 C8                   INY            ;NEXT BYTE
92E1 C0 40                CPY  #64
92E3 D0 F9                BNE  LOOP3     ;NOT FINISHED
92E5 A0 00                LDY  #0        ;CLEAR COLOR (TEXT) SCREEN
92E7 84 FD                STY  MISC      ;ZERO HIGH BYTE
92E9 A5 3D                LDA  PLCOL     ;GET CURRENT PLOT COLOR
92EB 0A                   ASL  A         ;SHIFT OVER 4 TIMES (* 16)
92EC 0A                   ASL  A
92ED 0A                   ASL  A
92EE 0A                   ASL  A
92EF 4D 21 D0             EOR  VIC+33    ;ADD IN CURRENT BACKGROUND COLOR
92F2 29 F0                AND  #%11110000
92F4 4D 21 D0             EOR  VIC+33    ;LO = VIC+33, HI = PLCOL
92F7 A2 8C                LDX  #>CRT     ;HIGH BYTE
92F9 86 FE    COL1        STX  MISC+1
92FB 91 FD    COLCLR      STA  (MISC),Y   ;STORE COLOR ON SCREEN
```

```
92FD C8                 INY               ;NEXT BYTE
92FE D0 FB              BNE   COLCLR
9300 E8                 INX               ;NEXT PAGE
9301 E0 8F              CPX   #>3*256+CRT   ;3 PAGES ONLY
9303 D0 F4              BNE   COL1        ;NOT LAST PAGE
9305 86 FE              STX   MISC+1
9307 91 FD     COL2     STA   (MISC),Y    ;ON LAST PAGE
9309 C8                 INY
930A C0 E8              CPY   #232        ;232 BYTES ON LAST PAGE
930C D0 F9              BNE   COL2
930E 4C BC 93           JMP   JOYSTK
               ;
9311 C9 12     RVSCK    CMP   #18         ;CONTROL-R (RVS ON)
9313 D0 27              BNE   ENDCK
9315 A0 00              LDY   #0
9317 84 FB              STY   AD          ;ZERO LOW BYTE
9319 A9 A0              LDA   #>HIPAGE
931B 85 FC              STA   AD+1        ;SET HIGH BYTE
931D B1 FB     RVSLP1   LDA   (AD),Y
931F 49 FF              EOR   #$FF        ;FLIP ALL BITS
9321 91 FB              STA   (AD),Y      ;RETURN IT TO PAGE
9323 C8                 INY
9324 D0 F7              BNE   RVSLP1
9326 E6 FC              INC   AD+1
9328 A5 FC              LDA   AD+1        ;CHECK LAST PAGE
932A C9 BF              CMP   #>31*256+HIPAGE
932C D0 EF              BNE   RVSLP1      ;NOT YET
932E B1 FB     RVSLP2   LDA   (AD),Y
9330 49 FF              EOR   #$FF        ;FLIP LAST 64 BYTES
9332 91 FB              STA   (AD),Y
9334 C8                 INY
9335 C0 40              CPY   #64
9337 D0 F5              BNE   RVSLP2
9339 4C BC 93           JMP   JOYSTK
               ;
933C C9 06     ENDCK    CMP   #6          ;CONTROL-BACK ARROW
933E D0 31              BNE   LSCK
9340 A9 1B              LDA   #27         ;TURN OFF HIRES
9342 8D 11 D0           STA   VIC+17
9345 A9 15              LDA   #21         ;RESTORE SCREEN/CHARACTER SET
9347 8D 18 D0           STA   VIC+24
934A A9 08              LDA   #8          ;TURN OFF MULTICOLOR (IF ON)
934C 8D 16 D0           STA   VIC+22
934F A9 00              LDA   #0
9351 8D 15 D0           STA   VIC+21      ;TURN OFF SPRITES
9354 A9 C7              LDA   #196+3      ;VIC BANK 0
9356 8D 00 DD           STA   VBANK
9359 A9 37              LDA   #54+1       ;FLIP BASIC BACK IN
935B 85 01              STA   R6510
935D 20 B2 98           JSR   RSTOFF      ;DISABLE RASTER INTERRUPTS
9360 A0 2D              LDY   #45         ;READ ZERO PAGE BACK IN
9362 B9 00 CB OFFZER    LDA   ZERSTO,Y    ;READ FROM $CB00
9365 99 39 00           STA   57,Y
9368 A9 00              LDA   #0
936A 99 00 02           STA   INBUFF,Y    ;SIMULTANEOUSLY CLEAR LINE BUFFER
936D 88                 DEY
936E 10 F2              BPL   OFFZER
9370 60                 RTS               ;RETURN TO BASIC
               ;
9371 C9 40     LSCK     CMP   #"@ ;LOAD OR SAVE
9373 D0 03              BNE   COLRCK
9375 4C 2A 99           JMP   LS          ;GO DO INPUT/OUTPUT

               ;
               ;CHECK MULTIPLE-KEY FUNCTIONS
               ;
9378 A2 0F     COLRCK   LDX   #15         ;TOP OF TABLE
937A DD B5 9B LOOPC     CMP   COLORS,X    ;CHECK IF A COLOR
```

```
937D D0 08              BNE  NEXTC     ;NO
937F 86 3D              STX  PLCOL     ;IN TABLE SO SAVE IT
9381 8E 27 D0           STX  VIC+39    ;CHANGE SPRITE COLOR
9384 4C BC 93           JMP  JOYSTK
9387 CA        NEXTC    DEX            ;CHECK NEXT COLOR
9388 10 F0              BPL  LOOPC     ;IF NOT FINISHED
               ;
938A A2 03              LDX  #3        ;TABLE TOP FOR FUNCTION KEYS
938C DD C5 9B  LOOPF    CMP  FNCTNS,X   ;CHECK IF A FUNCTION KEY
938F D0 1B              BNE  NEXTF
9391 A5 3C              LDA  SCRNMD    ;CHECK HIRES/MULTICOLOR
9393 D0 06              BNE  SETX      ;IF MULTICOLOR
9395 E0 03              CPX  #3        ;CHECK FOR F1
9397 D0 02              BNE  SETX      ;NO
9399 A2 01              LDX  #1        ;YES, SO SET IT F5
939B 86 3B     SETX     STX  PLOTMD    ;SET PLOT MODE
939D A2 08              LDX  #8        ;DEFAULT TYPE
939F A5 3B              LDA  PLOTMD
93A1 F0 04              BEQ  SETT      ;IF ERASE
93A3 0A                 ASL  A         ;SHIFT OVER 3 TIMES ...
93A4 0A                 ASL  A
93A5 0A                 ASL  A
93A6 AA                 TAX            ;... AND STORE IN .X
93A7 86 46     SETT     STX  TYPE      ;PUT IN TYPE
93A9 4C BC 93           JMP  JOYSTK
93AC CA        NEXTF    DEX            ;NEXT FUNCTION KEY
93AD 10 DD              BPL  LOOPF
               ;
93AF 38                 SEC            ;CHECK IF A NUMBER 1 TO 9
93B0 29 EF              AND  #239      ;TO INCLUDE SHIFTED NUMBERS
93B2 E9 20              SBC  #32       ;TO MAKE IT HEX 0-9
93B4 F0 06              BEQ  JOYSTK    ;A "0" NO GOOD
93B6 C9 0A              CMP  #10       ;AN ASCII ":" OR GREATER
93B8 B0 02              BCS  JOYSTK
93BA 85 3E              STA  PLINC     ;NEW PLOT INCREMENT

               ;
               ;CHECK THE JOYSTICK
               ;
93BC A5 A2     JOYSTK   LDA  TIME+2    ;CURRENT JIFFY
93BE C5 41              CMP  TIMSTO    ;LAST RECORDED VALUE
93C0 D0 03              BNE  GETJOY    ;ONLY ONCE/JIFFY
93C2 4C 5E 94           JMP  PLOT
93C5 A5 A2     GETJOY   LDA  TIME+2    ;JIFFY COUNTER
93C7 85 41              STA  TIMSTO    ;RESET WAIT LOOP
93C9 AD 00 DC           LDA  JSTICK
93CC 49 7F              EOR  #%01111111   ;SET PUSHED = 1
93CE 85 42              STA  JOY       ;SAVE FOR FUTURE REFERENCE
93D0 A5 C5              LDA  LSTX      ;CHECK KEY PUSHED
93D2 C9 05              CMP  #5        ;IF NOT F3
93D4 D0 03              BNE  FIRECK    ;THEN CHECK JOYSTICK
93D6 4C 5E 94           JMP  PLOT      ;ELSE SKIP
93D9 A5 42     FIRECK   LDA  JOY
93DB 29 10              AND  #%00010000   ;CHECK FOR FIRE BUTTON
93DD F0 19              BEQ  SETPR     ;NOT PRESSED
93DF A5 3A              LDA  LINES
93E1 F0 06              BEQ  BNCECK    ;NO LINE MODE
93E3 20 38 95           JSR  LINE      ;DRAW  A LINE
93E6 4C F8 93           JMP  SETPR
93E9 A5 40     BNCECK   LDA  PRESSD    ;IF JUST PRESSED
93EB D0 0F              BNE  CHECKU
93ED E6 40              INC  PRESSD    ;SET AS JUST PRESSED
93EF A5 39              LDA  DRAW
93F1 49 01              EOR  #%00000001   ;TOGGLE
93F3 85 39              STA  DRAW
93F5 4C FC 93           JMP  CHECKU    ;SKIP RESET
93F8 A9 00     SETPR    LDA  #0
93FA 85 40              STA  PRESSD    ;NOT JUST PRESSED
```

```
                        ;
93FC A5 42      CHECKU  LDA  JOY
93FE 29 01              AND  #%00000001   ;UP
9400 F0 0B              BEQ  CHECKD     ;NOT PRESSED
9402 A5 45              LDA  YPOS       ;DECREMENT YPOS
9404 38                 SEC
9405 E5 3E              SBC  PLINC
9407 C9 C8              CMP  #200       ;WRAP-AROUND ILLEGAL
9409 B0 02              BCS  CHECKD
940B 85 45              STA  YPOS       ;LEGAL, SO STORE IT
                        ;
940D A5 42      CHECKD  LDA  JOY
940F 29 02              AND  #%00000010   ;DOWN
9411 F0 0B              BEQ  CHECKL     ;NOT PRESSED
9413 A5 45              LDA  YPOS       ;INCREMENT YPOS
9415 18                 CLC
9416 65 3E              ADC  PLINC
9418 C9 C8              CMP  #200       ;CHECK IF YPOS>=200
941A B0 02              BCS  CHECKL     ;IT IS
941C 85 45              STA  YPOS       ;NO, SO STORE IT
                        ;
941E A5 42      CHECKL  LDA  JOY
9420 29 04              AND  #%00000100   ;LEFT
9422 F0 17              BEQ  CHECKR     ;PRESSED
9424 A5 43              LDA  XPOS       ;DECREMENT LOW BYTE
9426 38                 SEC
9427 E5 3E              SBC  PLINC
9429 85 FD              STA  MISC       ;TEMP SAVE
942B A5 44              LDA  XPOS+1     ;DECREMENT HIGH IF NECESSARY
942D E9 00              SBC  #0
942F 85 FE              STA  MISC+1
9431 30 08              BMI  CHECKR     ;IF HIGH BYTE NEGATIVE, ILLEGAL
9433 A5 FD              LDA  MISC       ;ELSE STORE MISCS
9435 85 43              STA  XPOS
9437 A5 FE              LDA  MISC+1
9439 85 44              STA  XPOS+1
                        ;
943B A5 42      CHECKR  LDA  JOY
943D 29 08              AND  #%00001000   ;RIGHT
943F F0 1D              BEQ  PLOT       ;NOT PRESSED
9441 A5 43              LDA  XPOS       ;INCREMENT LOW BYTE
9443 18                 CLC
9444 65 3E              ADC  PLINC
9446 85 FD              STA  MISC       ;TEMPORARY STORAGE
9448 A5 44              LDA  XPOS+1     ;INCREMENT HIGH IF NECESSARY
944A 69 00              ADC  #0
944C 85 FE              STA  MISC+1
944E F0 06              BEQ  DOR        ;LEGAL
9450 A5 FD              LDA  MISC
9452 C9 40              CMP  #64        ;FAR RIGHT EDGE
9454 B0 08              BCS  PLOT       ;ILLEGAL
9456 A5 FD      DOR     LDA  MISC       ;LEGAL, SO STORE
9458 85 43              STA  XPOS
945A A5 FE              LDA  MISC+1
945C 85 44              STA  XPOS+1

                        ;
                        ;MAIN PLOTTING ROUTINE
                        ;
945E A5 43      PLOT    LDA  XPOS       ;POSITION SPRITE
9460 A6 3C              LDX  SCRNMD
9462 F0 02              BEQ  OFFSET     ;IF NOT MULTICOLOR
9464 29 FE              AND  #254       ;HALF HORIZONTAL RESOLUTION
9466 18         OFFSET  CLC
9467 69 0D              ADC  #13        ;HORIZONTAL OFFSET
9469 8D 00 D0           STA  VIC        ;SPRITE 0 X REGISTER
946C A5 44              LDA  XPOS+1     ;HIGH BYTE
946E 69 00              ADC  #0
```

```
9470 8D 10 D0            STA  VIC+16  ;SPRITE 0'S BIT 8
9473 A5 45               LDA  YPOS
9475 69 2B               ADC  #43     ;VERTICAL OFFSET
9477 8D 01 D0            STA  VIC+1   ;SPRITE 0 Y REGISTER
947A A5 39               LDA  DRAW    ;CHECK IF JUST MOVING
947C D0 03               BNE  DOPL    ;NO
947E 4C 67 91            JMP  BEGIN   ;IF DRAW = 0
9481 A5 3C      DOPL     LDA  SCRNMD
9483 D0 0C               BNE  DOPL1   ;IF IN MULTICOLOR
9485 A5 3B               LDA  PLOTMD
9487 C9 02               CMP  #2
9489 D0 06               BNE  DOPL1   ;IF NOT IN F3-MODE
948B 20 E7 94            JSR  COLPUT  ;SKIP BIT-PLOTTING
948E 4C 67 91            JMP  BEGIN   ;RETURN TO LOOP
9491 20 C2 94   DOPL1    JSR  PLOTIT
9494 4C 67 91            JMP  BEGIN
                ;
9497 A5 45      LOC      LDA  YPOS    ;FIND ADDRESS IN AD
9499 4A                  LSR  A
949A 4A                  LSR  A
949B 4A                  LSR  A       ;DIVIDE BY 8
949C AA                  TAX          ;SHIFT TO .X FOR LATER USE
949D A5 43               LDA  XPOS    ;SET UP LOW YPOS HIGH XPOS
949F 45 45               EOR  YPOS
94A1 29 F8               AND  #%11111000
94A3 45 45               EOR  YPOS
94A5 18                  CLC
94A6 7D 80 CB            ADC  LINEL,X  ;ADD LO-BYTE LINE ADDRESSES
94A9 85 FB               STA  AD      ;STORE IT
94AB A5 44               LDA  XPOS+1  ;GET HIGH BYTE
94AD 7D A0 CB            ADC  LINEH,X  ;ADD HIGH-BYTE ADDRESSES
94B0 85 FC               STA  AD+1    ;STORE IT
94B2 A5 43               LDA  XPOS
94B4 29 07               AND  #7      ;GET LOW 3 BITS
94B6 A6 3C               LDX  SCRNMD
94B8 F0 04               BEQ  TRANS   ;IF HIRES
94BA 29 FE               AND  #254
94BC 05 46               ORA  TYPE    ;ADD ON TABLE OFFSET FOR PIXEL/MASK
94BE AA         TRANS    TAX          ;LEAVE TABLE POINTER IN .X
94BF A0 00               LDY  #0      ;ZERO .Y FOR USE
94C1 60                  RTS          ;END OF SUBROUTINE
                ;
94C2 20 97 94   PLOTIT   JSR  LOC     ;USE ABOVE ROUTINE FOR AD
94C5 A5 4F               LDA  XORIT
94C7 D0 0C               BNE  XOR     ;IF RUBBERBAND
94C9 A5 3B               LDA  PLOTMD
94CB D0 10               BNE  ADD     ;IF NOT F7
94CD B1 FB               LDA  (AD),Y  ;GET FROM HIRES SCREEN
94CF 3D E0 CB            AND  MASK,X  ;AND OFF ALL BUT PIXEL
94D2 4C E5 94            JMP  POKE    ;GO AND LOAD IT BACK
94D5 B1 FB      XOR      LDA  (AD),Y
94D7 5D C0 CB            EOR  PIXEL,X  ;FLIP THAT PIXEL
94DA 4C E5 94            JMP  POKE
94DD B1 FB      ADD      LDA  (AD),Y
94DF 3D E0 CB            AND  MASK,X  ;CLEAR THE SPACE
94E2 1D C0 CB            ORA  PIXEL,X  ;LOAD IN THE CORRECT PIXEL
94E5 91 FB      POKE     STA  (AD),Y  ;BACK ON THE SCREEN
                ;
                ;POKE CORRECT COLOR VALUE
                ;
94E7 A5 3B      COLPUT   LDA  PLOTMD
94E9 D0 01               BNE  DOCOL   ;IF NOT F7
94EB 60                  RTS          ;NO, DON'T RESET COLORS
94EC A5 45      DOCOL    LDA  YPOS
94EE 4A                  LSR  A       ;GET YPOS/8
94EF 4A                  LSR  A
```

```
94F0 4A             LSR  A
94F1 A8             TAY              ;USE AS POINTER TO 40'S TABLE
94F2 A5 44          LDA  XPOS+1
94F4 4A             LSR  A           ;GET BIT FROM HIGH BYTE
94F5 A5 43          LDA  XPOS        ;LOAD LOW BYTE
94F7 6A             ROR  A           ;GET THE HIGH BIT IN
94F8 4A             LSR  A           ;SHIFT TWICE
94F9 4A             LSR  A
94FA 18             CLC
94FB 79 40 CB       ADC  FORTYL,Y  ;ADD ON LOW BYTE
94FE 85 FD          STA  MISC     ;MISC HOLDS COLOR SCREEN ADDRESS
9500 B9 60 CB       LDA  FORTYH,Y  ;AND HIGH BYTE
9503 69 00          ADC  #0          ;IF SPILLOVER
9505 85 FE          STA  MISC+1
9507 A0 00          LDY  #0          ;FOR LATER USE
              ;
9509 A5 3B    MODE1  LDA  PLOTMD
950B C9 01          CMP  #%01     ;F5
950D D0 11          BNE  MODE2
950F B1 FD    DO1   LDA  (MISC),Y
9511 29 0F          AND  #%00001111  ;TAKE LOW NYBBLE
9513 85 FB          STA  AD          ;TEMP
9515 A5 3D          LDA  PLCOL
9517 0A             ASL  A           ;PUSH COLOR INTO HIGH NYBBLE
9518 0A             ASL  A
9519 0A             ASL  A
951A 0A             ASL  A
951B 05 FB          ORA  AD          ;PUT TOGETHER ..
951D 91 FD          STA  (MISC),Y   ;AND STORE AT COLOR SCREEN
951F 60             RTS
              ;
9520 C9 02    MODE2  CMP  #%10     ;IF F3
9522 D0 09          BNE  MODE3
9524 B1 FD          LDA  (MISC),Y
9526 29 F0          AND  #%11110000  ;CLEAR LOW NYBBLE
9528 05 3D          ORA  PLCOL    ;ADD COLOR IN
952A 91 FD          STA  (MISC),Y   ;STORE
952C 60             RTS
              ;
952D A5 FE    MODE3  LDA  MISC+1   ;TO COLOR SCREEN
952F 49 54          EOR  #%01010100  ;CHANGE HIGH BYTE TO $D800
9531 85 FE          STA  MISC+1
9533 A5 3D          LDA  PLCOL
9535 91 FD          STA  (MISC),Y   ;ON COLOR SCREEN
9537 60             RTS
              ;
              ;SUBROUTINE TO DRAW A LINE
              ;
9538 A5 43    LINE   LDA  XPOS     ;TRANSFER CURRENT TO END
953A 85 54          STA  X2
953C A5 44          LDA  XPOS+1
953E 85 55          STA  X2+1
9540 A5 45          LDA  YPOS
9542 85 56          STA  Y2
9544 20 6A 95       JSR  LINEDO   ;EXECUTE THE LINE
9547 A5 54          LDA  X2
9549 85 43          STA  XPOS     ;TRANSFER IT BACK
954B A5 55          LDA  X2+1
954D 85 44          STA  XPOS+1
954F A5 56          LDA  Y2
9551 85 45          STA  YPOS
9553 A5 3A          LDA  LINES
9555 C9 02          CMP  #2
9557 D0 10          BNE  FINLIN   ;IF NOT A DRAWTO
9559 A5 4F          LDA  XORIT
955B D0 0C          BNE  FINLIN   ;IF A RUBBERBAND
```

```
955D A5 54            LDA  X2
955F 85 51            STA  X1         ;OTHERWISE, SET A NEW BEGINNING
9561 A5 55            LDA  X2+1
9563 85 52            STA  X1+1
9565 A5 56            LDA  Y2
9567 85 53            STA  Y1
9569 60      FINLIN   RTS             ;END OF CONTROL LOOP
             ;
956A A5 3A   LINEDO   LDA  LINES
956C C9 03            CMP  #3
956E D0 2B            BNE  LINER      ;IF NOT A RIGHT-DRAW
9570 A5 3C            LDA  SCRNMD
9572 F0 06            BEQ  RGT        ;IF HIRES
9574 A5 43            LDA  XPOS
9576 29 FE            AND  #254       ;ONLY EVEN POSITIONS
9578 85 43            STA  XPOS
957A A5 43   RGT      LDA  XPOS       ;INCREMENT OUR X-POSITION
957C 18               CLC
957D 65 4A            ADC  FLLINC
957F 85 43            STA  XPOS
9581 90 02            BCC  RGT2       ;CARRY CLEAR IF NO OVERFLOW
9583 E6 44            INC  XPOS+1     ;OTHERWISE INCREMENT HIGH BYTE
9585 A5 44   RGT2     LDA  XPOS+1
9587 F0 06            BEQ  SIDE       ;NOT TO RIGHT SIDE YET
9589 A5 43            LDA  XPOS
958B C9 40            CMP  #64
958D B0 0B            BCS  ENDIT      ;IF AT RIGHT EDGE
958F 20 82 97 SIDE    JSR  PEEK       ;CHECK X,Y POSITION
9592 F0 06            BEQ  ENDIT      ;IF A STOP PATTERN
9594 20 C2 94         JSR  PLOTIT     ;PLOT THE POINT
9597 4C 7A 95         JMP  RGT        ;REPEAT THE LOOP
959A 60      ENDIT    RTS
             ;
959B A5 54   LINER    LDA  X2         ;DEFAULT XDIFF,YDIFF
959D 38               SEC             ;SET XDIFF = X2 - X1
959E E5 51            SBC  X1
95A0 85 57            STA  XDIFF
95A2 A5 55            LDA  X2+1       ;HIGH BYTES
95A4 E5 52            SBC  X1+1
95A6 85 58            STA  XDIFF+1
95A8 A5 56            LDA  Y2         ;AND SET YDIFF = Y2 - Y1
95AA 38               SEC
95AB E5 53            SBC  Y1
95AD 85 59            STA  YDIFF


95AF A0 01            LDY  #1         ;FIND SIGNUM OF X'S
95B1 A2 00            LDX  #0         ;DEFAULT TO $0001 (IN .X,.Y)
95B3 A5 52            LDA  X1+1
95B5 C5 55            CMP  X2+1
95B7 90 19            BCC  SGNY       ;IF HIGH BYTE X2<X1 THEN OK
95B9 D0 06            BNE  CHX        ;IF NOT EQUAL, CHANGE SIGNUM
95BB A5 54            LDA  X2         ;CHECK LOW BYTES
95BD C5 51            CMP  X1
95BF B0 11            BCS  SGNY       ;IF X1>=X2 THEN OK
95C1 A0 FF   CHX      LDY  #$FF       ;CHANGE SIGNUM TO $FFFF
95C3 A2 FF            LDX  #$FF
95C5 A5 51            LDA  X1         ;AND MAKE XDIFF = X1 - X2
95C7 38               SEC
95C8 E5 54            SBC  X2
95CA 85 57            STA  XDIFF
95CC A5 52            LDA  X1+1       ;HIGH BYTES
95CE E5 55            SBC  X2+1
95D0 85 58            STA  XDIFF+1
95D2 84 64   SGNY     STY  XSGN       ;STORE NEW SIGNUM
95D4 86 65            STX  XSGN+1
```

```
95D6 A0 01              LDY  #1        ;FIND SIGNUM OF Y'S
95D8 A5 56              LDA  Y2
95DA C5 53              CMP  Y1
95DC B0 09              BCS  ABSX      ;IF Y1>=Y2 THEN OK
95DE A0 FF              LDY  #$FF      ;ELSE CHANGE SIGNS
95E0 A5 53              LDA  Y1        ;AND MAKE YDIFF = Y1 - Y2
95E2 38                 SEC
95E3 E5 56              SBC  Y2
95E5 85 59              STA  YDIFF
95E7 84 66     ABSX     STY  YSGN      ;STORE Y SIGNUM
95E9 A9 00              LDA  #0        ;ZERO X & Y COUNTER
95EB 85 62              STA  YCNTR
95ED 85 60              STA  XCNTR
               ;
95EF A6 57     DISFND   LDX  XDIFF     ;FIND GREATER DIS
95F1 A4 58              LDY  XDIFF+1
95F3 D0 0E              BNE  XSET      ;IF HIGH BYTE X SET, GREATER THAN Y
95F5 E4 59              CPX  YDIFF     ;ELSE COMPARE LOW BYTES
95F7 B0 0A              BCS  XSET      ;IF YDIFF>=XDIFF, SET YDIFF GREATER
95F9 A6 59              LDX  YDIFF
95FB 20 0B 96           JSR  HAFDIS    ;PUT .Y,.X IN DIS
95FE 85 60              STA  XCNTR     ;.A HOLDS HALF DISTANCE
9600 4C 14 96           JMP  INIT
9603 20 0B 96  XSET     JSR  HAFDIS
9606 85 62              STA  YCNTR     ;STORE DIS/2 IN YCNTR
9608 4C 14 96           JMP  INIT
960B 84 5B     HAFDIS   STY  DIS+1     ;HIGH BYTE
960D 98                 TYA            ;GET THE CARRY BIT
960E 4A                 LSR  A
960F 86 5A              STX  DIS       ;LOW BYTE
9611 8A                 TXA
9612 6A                 ROR  A         ;CARRY BYTE IN AT LEFT
9613 60                 RTS            ;RETURN 1/2 DIS IN .A
               ;
9614 A9 00     INIT     LDA  #0        ;INITIALIZE VARIABLES
9616 85 5E              STA  CNTR      ;ZERO DISTANCE COUNTER
9618 85 5F              STA  CNTR+1
961A 85 61              STA  XCNTR+1    ;ZERO HIGH BYTES X,Y COUNTERS
961C 85 63              STA  YCNTR+1
961E A5 51              LDA  X1        ;INITIALIZE XPOS,YPOS
9620 85 43              STA  XPOS
9622 A5 52              LDA  X1+1
9624 85 44              STA  XPOS+1
9626 A5 53              LDA  Y1
9628 85 45              STA  YPOS
962A A5 5A              LDA  DIS       ;SET UP DIS2
962C 18                 CLC
962D 69 01              ADC  #1
962F 85 5C              STA  DIS2
9631 A5 5B              LDA  DIS+1     ;HIGH BYTES
9633 69 00              ADC  #0
9635 85 5D              STA  DIS2+1
               ;
9637 A5 3C     LOOP     LDA  SCRNMD
9639 F0 0E              BEQ  LOOPDO    ;IF IN HIRES MODE
963B A5 FE              LDA  MISC+1
963D C5 45              CMP  YPOS
963F D0 08              BNE  LOOPDO    ;MISC+1 HOLDS OLD YPOS
9641 A5 43              LDA  XPOS
9643 29 FE              AND  #$FE      ;EVEN COLUMNS ONLY
9645 C5 FD              CMP  MISC
9647 F0 03              BEQ  XINC      ;MISC HOLDS OLD XPOS
9649 20 C2 94  LOOPDO   JSR  PLOTIT    ;PLOT POINT
               ;
964C A5 43     XINC     LDA  XPOS
964E 29 FE              AND  #$FE
9650 85 FD              STA  MISC      ;SAVE MISC
```

```
9652 A5 45              LDA  YPOS
9654 85 FE              STA  MISC+1   ;SAVE YPOS
9656 A5 60              LDA  XCNTR    ;ADD XDIFF TO XCNTR
9658 18                 CLC
9659 65 57              ADC  XDIFF
965B 85 60              STA  XCNTR
965D A5 61              LDA  XCNTR+1   ;HIGH BYTES
965F 65 58              ADC  XDIFF+1
9661 85 61              STA  XCNTR+1
9663 C5 5B              CMP  DIS+1
9665 F0 04              BEQ  LOXCK    ;CHECK LOW BYTES
9667 90 21              BCC  YGREAT   ;NO INCREMENT TO XPOS
9669 D0 06              BNE  XDO      ;GO INCREMENT XPOS
966B A5 60      LOXCK   LDA  XCNTR
966D C5 5A              CMP  DIS
966F 90 19              BCC  YGREAT   ;IGNORE IF DIS<XCNTR
9671 A5 60      XDO     LDA  XCNTR    ;PULL DIS OFF XCNTR
9673 E5 5A              SBC  DIS
9675 85 60              STA  XCNTR
9677 A5 61              LDA  XCNTR+1   ;HIGH BYTES
9679 E5 5B              SBC  DIS+1
967B 85 61              STA  XCNTR+1
967D A5 43              LDA  XPOS     ;ADD XSGN (1 OR -1) TO YPOS
967F 18                 CLC
9680 65 64              ADC  XSGN
9682 85 43              STA  XPOS
9684 A5 44              LDA  XPOS+1
9686 65 65              ADC  XSGN+1
9688 85 44              STA  XPOS+1
                ;
968A A5 62      YGREAT  LDA  YCNTR    ;ADD YDIFF
968C 18                 CLC
968D 65 59              ADC  YDIFF
968F 85 62              STA  YCNTR
9691 A5 63              LDA  YCNTR+1   ;HIGH BYTES
9693 69 00              ADC  #0
9695 85 63              STA  YCNTR+1
9697 C5 5B              CMP  DIS+1
9699 F0 04              BEQ  LOYCK    ;CHECK LOW BYTES
969B 90 1B              BCC  CINC     ;DON'T CHANGE YPOS
969D D0 06              BNE  YDEC     ;CHANGE YPOS
969F A5 62      LOYCK   LDA  YCNTR
96A1 C5 5A              CMP  DIS
96A3 90 13              BCC  CINC     ;IGNORE IF LESS
96A5 A5 62      YDEC    LDA  YCNTR    ;PULL DIS FROM YCNTR
96A7 E5 5A              SBC  DIS      ;CARRY SET
96A9 85 62              STA  YCNTR
96AB A5 63              LDA  YCNTR+1   ;HIGH BYTES
96AD E5 5B              SBC  DIS+1
96AF 85 63              STA  YCNTR+1
96B1 A5 45              LDA  YPOS     ;ADD YSGN (1 OR -1) TO YPOS
96B3 18                 CLC
96B4 65 66              ADC  YSGN
96B6 85 45              STA  YPOS
                ;
96B8 E6 5E      CINC    INC  CNTR     ;INCREMENT COUNTER
96BA D0 02              BNE  CINC2    ;IF NO OVERFLOW
96BC E6 5F              INC  CNTR+1   ;INCREMENT HIGH BYTE
96BE A5 5F      CINC2   LDA  CNTR+1
96C0 C5 5D              CMP  DIS2+1
96C2 90 06              BCC  CONT     ;NOT FINISHED YET
96C4 A5 5E              LDA  CNTR
96C6 C5 5C              CMP  DIS2
96C8 B0 03              BCS  RETURN   ;FINISHED
96CA 4C 37 96   CONT    JMP  LOOP     ;BACK TO LOOP
96CD 60         RETURN  RTS
                ;
```

```
                 ;SUBROUTINE TO FILL AN AREA
                 ;
96CE A9 00      FILL     LDA  #0
96D0 85 3F               STA  PNTR      ;BOTTOM OF STACK
96D2 A5 3C               LDA  SCRNMD
96D4 F0 06               BEQ  BTFIND    ;IF HIRES
96D6 A5 43               LDA  XPOS
96D8 29 FE               AND  #$FE
96DA 85 43               STA  XPOS      ;SET TO AN EVEN COLUMN
96DC A9 00      BTFIND   LDA  #0        ;CLEAR UP/DOWN FLAGS

96DE 85 48               STA  DOWNF
96E0 85 47               STA  UPF
96E2 A5 44      FIND     LDA  XPOS+1
96E4 D0 04               BNE  DOFIND    ;STILL ON 256-319
96E6 A5 43               LDA  XPOS
96E8 F0 1F               BEQ  FILLBT    ;AT LEFT EDGE
96EA A5 43      DOFIND   LDA  XPOS      ;DECREMENT XPOS
96EC 38                  SEC
96ED E5 4A               SBC  FLLINC
96EF 85 43               STA  XPOS
96F1 A5 44               LDA  XPOS+1    ;HIGH BYTES
96F3 E9 00               SBC  #0
96F5 85 44               STA  XPOS+1
96F7 20 82 97            JSR  PEEK
96FA D0 E6               BNE  FIND      ;KEEP SCANNING LEFT
96FC A5 43               LDA  XPOS      ;BACK RIGHT ONE PIXEL
96FE 18                  CLC
96FF 65 4A               ADC  FLLINC
9701 85 43               STA  XPOS
9703 A5 44               LDA  XPOS+1
9705 69 00               ADC  #0
9707 85 44               STA  XPOS+1

                 ;
9709 E6 45      FILLBT   INC  YPOS      ;CHECK BELOW
970B 20 82 97            JSR  PEEK      ;GET A VALUE
970E F0 0D               BEQ  DZER      ;EQUAL TO PLOT, SO ...
9710 A5 48               LDA  DOWNF
9712 D0 0D               BNE  UCK       ;FLAG SET
9714 20 B2 97            JSR  PUSH      ;STORE THIS LOCATION
9717 A9 01               LDA  #1
9719 85 48               STA  DOWNF     ;SET THE FLAG
971B D0 04               BNE  UCK       ;BRANCH-ALWAYS
971D A9 00      DZER     LDA  #0
971F 85 48               STA  DOWNF     ;... RESET DOWN FLAG
9721 C6 45      UCK      DEC  YPOS
9723 C6 45               DEC  YPOS      ;LOOK ABOVE THE PLOT
9725 20 82 97            JSR  PEEK
9728 F0 0D               BEQ  UZER      ;IF THE SAME AS PLOT ...
972A A5 47               LDA  UPF

972C D0 0D               BNE  DOPOKE    ;FLAG SET
972E 20 B2 97            JSR  PUSH      ;SAVE THE LOCATION
9731 A9 01               LDA  #1
9733 85 47               STA  UPF       ;SET THE UP FLAG
9735 D0 04               BNE  DOPOKE    ;BRANCH-ALWAYS
9737 A9 00      UZER     LDA  #0
9739 85 47               STA  UPF       ;... CLEAR THE FLAG

973B E6 45      DOPOKE   INC  YPOS      ;RESET YPOS
973D 20 C2 94            JSR  PLOTIT    ;PLOT THIS POINT
9740 A5 43               LDA  XPOS      ;GO A PIXEL RIGHT
9742 18                  CLC
9743 65 4A               ADC  FLLINC
9745 85 43               STA  XPOS
9747 A5 44               LDA  XPOS+1    ;HIGH BYTES
9749 69 00               ADC  #0
974B 85 44               STA  XPOS+1
974D A5 44      PXCK     LDA  XPOS+1
```

```
974F F0 06             BEQ  ENDTST   ;NOT AT RIGHT EDGE
9751 A5 43             LDA  XPOS
9753 C9 40             CMP  #64
9755 B0 05             BCS  PULL     ;AT RIGHT EDGE
9757 20 82 97 ENDTST   JSR  PEEK
975A D0 AD             BNE  FILLBT   ;NOT TO A SAME PIXEL
975C A4 3F    PULL     LDY  PNTR
975E F0 65             BEQ  ENDPSH   ;NOTHING LEFT TO PULL
9760 20 E4 FF          JSR  GETIN
9763 C9 00             CMP  #0
9765 D0 5E             BNE  ENDPSH   ;KEY PRESSED = ABORT
9767 88       DOPULL   DEY
9768 B9 00 9D          LDA  TABLE1,Y  ;PULL Y,X,X+1 OFF TABLES
976B 85 45             STA  YPOS
976D B9 00 9E          LDA  TABLE2,Y
9770 85 44             STA  XPOS+1
9772 B9 00 9F          LDA  TABLE3,Y
9775 85 43             STA  XPOS
9777 84 3F             STY  PNTR     ;STORE RESET POINTER
9779 A5 45             LDA  YPOS
977B C9 C8             CMP  #200
977D B0 DD             BCS  PULL     ;IF OUT OF RANGE
977F 4C DC 96          JMP  BTFIND   ;RE-ENTER LOOP
              ;
9782 20 97 94 PEEK     JSR  LOC      ;GET ADDRESS AND .X
9785 BD E0 CB          LDA  MASK,X
9788 49 FF             EOR  #$FF
978A 31 FB             AND  (AD),Y   ;MASK OFF ALL BUT PIXELS
978C 48                PHA           ;PUSH ONTO STACK
978D 8A                TXA           ;AND X WITH #7
978E 29 07             AND  #7
9790 AA                TAX
9791 68                PLA           ;RECALL THE PIXEL PATTERN
9792 E4 49             CPX  FLSHFT
9794 B0 06             BCS  ENDPK    ;DON'T SHIFT
9796 4A       PEEKLP   LSR  A        ;SHIFT TOWARDS BOTTOM BITS
9797 E8                INX           ;INCREMENT SHIFT COUNTER
9798 E4 49             CPX  FLSHFT
979A 90 FA             BCC  PEEKLP   ;NOT FINISHED
979C A6 50    ENDPK    LDX  FILBOR
979E D0 05             BNE  ANY      ;ANY PIXEL WILL STOP THE FILL
97A0 C5 3B             CMP  PLOTMD   ;ZERO SET IF END OF LINE
97A2 4C B1 97          JMP  ENDPK2
97A5 A2 01    ANY      LDX  #1       ;CLEAR ZERO FLAG
97A7 08                PHP           ;SAVE THIS
97A8 C9 00             CMP  #0       ;CHECK FOR SOMETHING ON SCREEN
97AA F0 04             BEQ  NOTH     ;NOTHING
97AC 28                PLP           ;SOMETHING, SO PULL ...
97AD A2 00             LDX  #0       ;SET ZERO ...
97AF 60                RTS           ;AND RETURN
97B0 28       NOTH     PLP           ;PULL THE ZERO FLAG
97B1 60       ENDPK2   RTS           ;RETURN
              ;
97B2 A4 3F    PUSH     LDY  PNTR     ;FILL STACK POINTER
97B4 A5 43             LDA  XPOS
97B6 99 00 9F          STA  TABLE3,Y  ;PUSH XPOS
97B9 A5 44             LDA  XPOS+1
97BB 99 00 9E          STA  TABLE2,Y  ;PUSH HIGH BYTE XPOS
97BE A5 45             LDA  YPOS
97C0 99 00 9D          STA  TABLE1,Y  ;PUSH YPOS
97C3 E6 3F             INC  PNTR     ;MOVE UP POINTER
97C5 60       ENDPSH   RTS
              ;
              ;CHANGE BORDER BACKGROUND
              ;
97C6 20 15 98 BORBAK   JSR  KEYS     ;GET A DIRECTION
```

```
97C9 C9 FF              CMP  #$FF
97CB D0 01              BNE  CHECK     ;A DIRECTION OR NOTHING
97CD 60                 RTS            ;NOT RECOGNIZED BY KEYS
97CE C9 00     CHECK    CMP  #0
97D0 D0 07              BNE  EXEC      ;SOME DIRECTION WAS PRESSED
97D2 AD 00 DC           LDA  JSTICK
97D5 49 7F              EOR  #127
97D7 85 42              STA  JOY       ;GET A DIRECTION FROM JOYSTICK
97D9 A5 42     EXEC     LDA  JOY
97DB 29 10              AND  #16
97DD D0 35              BNE  ENDBB     ;FIRE BUTTON PRESSED
97DF A5 42              LDA  JOY
97E1 29 03              AND  #3
97E3 F0 10              BEQ  NEXT1     ;NO VERTICAL MOTION
97E5 0A                 ASL  A         ;DOUBLE
97E6 38                 SEC            ;SUBTRACT 3 (-1 OR 1 RESULTS)
97E7 E9 03              SBC  #3
97E9 49 FE              EOR  #$FE      ;TO GET 1 OR -1
97EB 18                 CLC            ;ADD BACKGROUND TO IT
97EC 6D 21 D0           ADC  VIC+33
97EF 8D 21 D0           STA  VIC+33
97F2 4C 06 98           JMP  DELIT

97F5 A5 42     NEXT1    LDA  JOY
97F7 29 0C              AND  #12
97F9 F0 CB              BEQ  BORBAK    ;NO HORIZONTAL MOTION
97FB 4A                 LSR  A         ;MAKE IT 2 OR 4
97FC 38                 SEC
97FD E9 03              SBC  #3        ;$FF OR $01
97FF 18                 CLC            ;ADD BORDER COLOR
9800 6D 20 D0           ADC  VIC+32
9803 8D 20 D0           STA  VIC+32    ;STORE IT
9806 A2 40     DELIT    LDX  #64       ;64 LONG LOOPS
9808 A0 FF              LDY  #$FF      ;ONE LONG LOOP

980A 48        DELIT1   PHA            ;DELAY
980B 68                 PLA
980C 88                 DEY
980D D0 FB              BNE  DELIT1
980F CA                 DEX
9810 D0 F8              BNE  DELIT1
9812 F0 B2              BEQ  BORBAK    ;BRANCH-ALWAYS
9814 60        ENDBB    RTS            ;RETURN TO MAIN LOOP

               ;
               ;GET DIR FROM KEYBOARD
               ;
9815 A5 C5     KEYS     LDA  LSTX
9817 C9 40              CMP  #64
9819 D0 05              BNE  CKKEY     ;SOMETHING IS BEING PRESSED
981B A9 00              LDA  #0        ;SET ZERO IN .A TO FLAG NO KEY
981D 85 42              STA  JOY
981F 60                 RTS
9820 A2 07     CKKEY    LDX  #7        ;DIRECTION-KEY TABLE
9822 DD A5 9B  KLOOP    CMP  DIRKEY,X  ;IN TABLE
9825 D0 06              BNE  KEND      ;GO TO LOOP END
9827 BD AD 9B           LDA  DIRECT,X  ;GET DIRECTION
982A 85 42              STA  JOY       ;STORE IT IN JOYSTICK
982C 60                 RTS
982D CA        KEND     DEX
982F 10 F2              BPL  KLOOP     ;CHECK 8 KEYS
9830 A9 00              LDA  #0        ;ZERO JOY
9832 85 42              STA  JOY
9834 A9 FF              LDA  #$FF      ;CLEAR ZERO FLAG
9836 60                 RTS            ;RETURN

               ;
               ;TOGGLE MESSAGE LINE
               ;
9837 A0 27     OUT      LDY  #39       ;COPY 40 BYTES
```

```
9839 B9 C0 8F XFER1     LDA  CRT+960,Y   ;FROM SCREEN
983C 99 40 BF           STA  SCLINE,Y   ;TO SCLINE TABLE
983F B9 C0 DB           LDA  COLCRT+960,Y   ;FROM COLOR SCREEN
9842 99 70 BF           STA  COLINE,Y   ;TO COLINE TABLE
9845 A5 3D              LDA  PLCOL    ;SET COLORS ON SCREEN
9847 99 C0 DB           STA  COLCRT+960,Y
984A A9 20              LDA  #32      ;CLEAR OUT OTHER DATA
984C 99 C0 8F           STA  CRT+960,Y
984F 88                 DEY
9850 10 E7              BPL  XFER1
9852 A9 1B              LDA  #27      ;STANDARD CHARACTERS
9854 85 4B              STA  HIR1     ;VIC+21 SHADOW
9856 A9 35              LDA  #53      ;UPPER CASE ROM CHARACTERS
9858 85 4C              STA  HIR2     ;VIC+24 SHADOW
985A A9 08              LDA  #8       ;NON-MULTICOLOR CHARACTERS
985C 85 4D              STA  HIR3     ;VIC+22 SHADOW
985E 60                 RTS
                   ;
985F A0 27        IN    LDY  #39      ;COPY 40 BYTES
9861 B9 40 BF XFER2     LDA  SCLINE,Y   ;FROM TABLE
9864 99 C0 8F           STA  CRT+960,Y   ;TO SCREEN
9867 B9 70 BF           LDA  COLINE,Y   ;FROM TABLE
986A 99 C0 DB           STA  COLCRT+960,Y   ;TO COLOR SCREEN
986D 88                 DEY
986E 10 F1              BPL  XFER2
9870 A9 3B              LDA  #59      ;HI-RES
9872 85 4B              STA  HIR1     ;VIC+21 SHADOW
9874 A9 38              LDA  #56      ;SECOND HI-RES SCREEN

9876 85 4C              STA  HIR2     ;VIC+24 SHADOW
9878 A5 4E              LDA  HIR4     ;OLD MULTICOLOR MODE
987A 85 4D              STA  HIR3     ;VIC+22 SHADOW
987C 60                 RTS


                   ;
                   ;ENABLE/DISABLE RASTER INTERRUPTS
                   ;
987D 78           RSTON SEI
987E A9 7F              LDA  #$7F
9880 8D 0D DC           STA  CIAICR   ;DISABLE CIA INTERRUPTS
9883 A9 01              LDA  #1
9885 8D 1A D0           STA  VIC+26   ;ENABLE RASTER INTERRUPTS
9888 A9 00              LDA  #0
988A 8D 12 D0           STA  VIC+18   ;SET IT AT LINE 0
988D AD 11 D0           LDA  VIC+17
9890 29 7F              AND  #127
9892 8D 11 D0           STA  VIC+17
9895 AD 14 03           LDA  INTPNT
9898 8D 22 99           STA  IEND+1   ;STORE THE OLD INTERRUPT
989B AD 15 03           LDA  INTPNT+1
989E 8D 23 99           STA  IEND+2
98A1 A9 D3              LDA  #<INT
98A3 8D 14 03           STA  INTPNT   ;VECTOR OUR INTERRUPT IN
98A6 A9 98              LDA  #>INT
98A8 8D 15 03           STA  INTPNT+1
98AB 58                 CLI           ;RE-ENABLE INTERRUPTS
98AC A9 01              LDA  #1
98AE 8D 15 D0           STA  VIC+21   ;TURN ON SPRITE 0
98B1 60                 RTS
                   ;
98B2 A9 00        RSTOFF LDA #0
98B4 8D 1A D0           STA  VIC+26   ;DISABLE RASTER INTERRUPTS
98B7 AD 0D DC           LDA  CIAICR
98BA 09 81              ORA  #129
98BC 8D 0D DC           STA  CIAICR   ;ENABLE CIA INTERRUPTS
98BF 78                 SEI
98C0 AD 22 99           LDA  IEND+1
98C3 8D 14 03           STA  INTPNT   ;REPOINT THE INTERRUPT
```

```
98C6 AD 23 99            LDA   IEND+2
98C9 8D 15 03            STA   INTPNT+1
98CC 58                  CLI            ;RE-ENABLE THE INTERRUPTS
98CD A9 00               LDA   #0
98CF 8D 15 D0            STA   VIC+21   ;TURN OFF THE SPRITES
98D2 60                  RTS

                ;
                ;RASTER INTERRUPT ROUTINE
                ;
98D3 AD 19 D0 INT        LDA   VIC+25   ;WHAT INTERRUPTS
98D6 8D 19 D0            STA   VIC+25   ;ACKNOWLEDGE THEM
98D9 29 01               AND   #1
98DB F0 47               BEQ   SKIP
98DD A5 4B               LDA   HIR1
98DF 8D 11 D0            STA   53265    ;SET DEFAULT HIRES
98E2 A5 4C               LDA   HIR2
98E4 8D 18 D0            STA   53272    ;AND CHARACTER SET
98E7 A5 4D               LDA   HIR3
98E9 8D 16 D0            STA   53270    ;AND MULTICOLOR
98EC A2 F2               LDX   #242
98EE A0 01               LDY   #1       ;SET .X TO RASTER LINE, FLAG .Y
98F0 AD 12 D0            LDA   VIC+18   ;CHECK RASTER FOR WHICH
98F3 10 04               BPL   MID
98F5 A2 00               LDX   #0
98F7 A0 00               LDY   #0       ;FLAG RASTER AT 0, FLAG .Y
98F9 8E 12 D0 MID        STX   VIC+18   ;SET NEXT INTERRUPT
98FC AD 11 D0            LDA   VIC+17
98FF 29 7F               AND   #127
9901 8D 11 D0            STA   VIC+17   ;KEEP BIT 8 ZERO
9904 C0 00               CPY   #0
9906 D0 03               BNE   NORMAL   ;RESET SCREEN
9908 4C 1A 99            JMP   INTCK    ;DEFAULTS CORRECT
990B A9 3B    NORMAL     LDA   #59
990D 8D 11 D0            STA   53265    ;HIRES
9910 A9 38               LDA   #56
9912 8D 18 D0            STA   53272    ;CHARACTERS
9915 A5 4E               LDA   HIR4
9917 8D 16 D0            STA   53270    ;MULTICOLOR
991A AD 0D DC INTCK      LDA   CIAICR
991D 29 01               AND   #1
991F F0 03               BEQ   SKIP     ;IF NO STANDARD INTERRUPT
9921 4C 31 EA IEND       JMP   INTRPT   ;JUMP TO NORMAL IRQ ROUTINE
9924 68       SKIP       PLA            ;ABORT THE INTERRUPT
9925 A8                  TAY
9926 68                  PLA
9927 AA                  TAX
9928 68                  PLA
9929 40                  RTI

                ;
                ;LOAD SAVE HIRES SUBROUTINE
                ;
992A 20 37 98 LS         JSR   OUT      ;BRING UP STATUS LINE
992D A2 FD               LDX   #<LSMESS
992F A0 99               LDY   #>LSMESS  ;SET INPUT ROUTINE PARAMETERS
9931 20 31 9A            JSR   INPUT    ;EXECUTE THE INPUT
9934 F0 48               BEQ   ENDLS    ;IF ZERO LENGTH, ABORT
9936 A9 00               LDA   #0
9938 85 02               STA   LDSV     ;DEFAULT (0=LOAD,1=SAVE)
993A AD 00 02            LDA   INBUFF   ;START OF INPUT RETURN BUFFER
993D C9 4C               CMP   #"L"
993F F0 06               BEQ   NXMES1   ;DEFAULT CORRECT
9941 C9 53    SCK        CMP   #"S"
9943 D0 39               BNE   ENDLS    ;NOT S OR L, SO ABORT
9945 E6 02               INC   LDSV     ;LDSV = 1 = SAVE
9947 A2 0B    NXMES1     LDX   #<DVMESS
9949 A0 9A               LDY   #>DVMESS  ;PARAMETERS
```

```
994B 20 31 9A            JSR  INPUT
994E C9 01               CMP  #1
9950 D0 2C               BNE  ENDLS    ;INPUT MUST BE 1 LONG
9952 AD 00 02            LDA  INBUFF
9955 38                  SEC
9956 E9 30               SBC  #"0"     ;SET ASCII "0" TO HEX 00
9958 A2 03               LDX  #3       ;CHECK SECONDARY ADDRESSES
995A DD 25 9A DVLOOP     CMP  DEV,X
995D F0 05               BEQ  DOLFS    ;FOUND DEVICE IN TABLE
995F CA                  DEX
9960 10 F8               BPL  DVLOOP   ;NEXT DEVICE
9962 30 1A               BMI  ENDLS    ;NOT IN TABLE
9964 BC 29 9A DOLFS      LDY  SA,X     ;.Y HOLDS SECONDARY ADDRESS
9967 85 3F               STA  PNTR     ;SAVE PHYSICAL DEVICE FOR REF.
9969 AA                  TAX           ;IN .X FOR OPEN FILE
996A E0 01               CPX  #1
996C D0 02               BNE  DOSET    ;NOT A TAPE
996E A4 02               LDY  LDSV     ;S.A. = 0 OR 1
9970 A9 01      DOSET    LDA  #1       ;SET LOGICAL DEVICE
9972 20 BA FF            JSR  SETLFS   ;SET LOGICAL,PHYSICAL,SECONDARY
9975 A2 1A               LDX  #<NMMESS
9977 A0 9A               LDY  #>NMMESS   ;INPUT "NAME OF FILE"
9979 20 31 9A            JSR  INPUT
997C D0 06               BNE  HERE     ;CONTINUE
997E 20 5F 98 ENDLS      JSR  IN
9981 4C 67 91            JMP  BEGIN    ;ABORT

9984 A5 3F      HERE     LDA  PNTR
9986 C9 08               CMP  #8
9988 90 12               BCC  NAMDO    ;NOT A DISK DRIVE
998A A5 02               LDA  LDSV
998C F0 0E               BEQ  NAMDO    ;NOT SAVING
998E A0 00               LDY  #0       ;INITIALIZE COUNTER
9990 B9 2D 9A SWADD      LDA  SW,Y
9993 9D 00 02            STA  INBUFF,X   ;.X SET FROM INPUT
9996 E8                  INX
9997 C8                  INY
9998 C0 04               CPY  #4
999A D0 F4               BNE  SWADD    ;ADD ON ",S,W"
999C 8A         NAMDO    TXA           ;.A HOLDS NAME LENGTH
999D A2 00               LDX  #<INBUFF   ;.X,.Y HOLDS INITIAL ADDRESS

999F A0 02               LDY  #>INBUFF
99A1 20 BD FF            JSR  SETNAM   ;SET FILE NAME
                ;
99A4 20 5F 98            JSR  IN       ;RESTORE FULL HIRES
99A7 20 B2 98            JSR  RSTOFF   ;DISABLE RASTER INTERRUPTS
99AA 20 95 9A            JSR  XFERIT   ;TRANSFER ALL DATA
99AD A9 01               LDA  #1       ;CLOSE FILE #1
99AF 20 C3 FF            JSR  CLOSE
99B2 20 7D 98            JSR  RSTON    ;RE-ENABLE RASTER INTERRUPTS
99B5 A5 3F               LDA  PNTR
99B7 C9 08               CMP  #8
99B9 90 3C               BCC  NODO     ;IF NOT A DISK
99BB 20 37 98            JSR  OUT      ;RESTORE STATUS LINE
99BE A9 0F               LDA  #15

99C0 A8                  TAY
99C1 A6 3F               LDX  PNTR
99C3 20 BA FF            JSR  SETLFS   ;SET 15,8,15
99C6 A9 00               LDA  #0
99C8 20 BD FF            JSR  SETNAM   ;NO NAME
99CB 20 C0 FF            JSR  OPEN
99CE A2 0F               LDX  #15
99D0 20 C6 FF            JSR  CHKIN    ;OPEN, AND SET FOR INPUT
99D3 A0 00               LDY  #0       ;POINTER TO SCREEN
99D5 20 CF FF ERLP       JSR  CHRIN    ;GET A BYTE
99D8 C9 0D               CMP  #13
99DA F0 0B               BEQ  WAITER   ;IF RETURN, FINISHED
```

```
99DC 29 3F              AND  #63       ;SET UP FOR POKING
99DE 99 C0 8F           STA  CRT+960,Y
99E1 C8                 INY            ;NEXT BYTE
99E2 20 B7 FF           JSR  READST    ;CHECK ERROR
99E5 F0 EE              BEQ  ERLP      ;NONE, SO BRANCH BACK
99E7 A9 0F     WAITER   LDA  #15
99E9 20 C3 FF           JSR  CLOSE     ;CLOSE 15
99EC A9 96              LDA  #150
99EE 85 A2              STA  TIME+2    ;SET A COUNT-UP TIME
99F0 A5 A2     WAITNG   LDA  TIME+2
99F2 D0 FC              BNE  WAITNG    ;WAIT 106 JIFFIES
99F4 20 5F 98           JSR  IN        ;FULL HIRES
99F7 20 E7 FF  NODO     JSR  CLALL     ;CLOSE EVERYTHING
99FA 4C 67 91           JMP  BEGIN     ;RETURN TO LOOP

               ;
               ;LOAD/SAVE, DEVICE, NAME QUERIES
99FD 0C 0F 01  LSMESS   .BYT 12,15,1,4,32,15,18,32,19,1,22,5,63,0
9A0B 04 05 16  DVMESS   .BYT 4,5,22,9,3,5,32,14,21,13,2,5,18,63,0
9A1A 06 09 0C  NMMESS   .BYT 6,9,12,5,32,14,1,13,5,58,0
9A25 01 02 08  DEV      .BYT 1,2,8,9   ;PHYSICAL DEVICES
9A29 01 00 02  SA       .BYT 1,0,2,2   ;SECONDARY ADDRESSES
9A2D 2C 53 2C  SW       .ASC ",S,W"   ;ADD-ON FOR DISK SAVE

               ;
9A31 86 FD     INPUT    STX  MISC
9A33 84 FE              STY  MISC+1    ;SAVE .X AND .Y
9A35 A0 27              LDY  #39
9A37 A9 20              LDA  #32       ;CLEAR STATUS LINE
9A39 99 C0 8F  CLRLN    STA  CRT+960,Y
9A3C 88                 DEY
9A3D 10 FA              BPL  CLRLN
9A3F C8                 INY            ;SAME AS LDY #0
9A40 B1 FD     PRINT    LDA  (MISC),Y
9A42 F0 06              BEQ  INPDO     ;IF END OF MESSAGE
9A44 99 C0 8F           STA  CRT+960,Y
9A47 C8                 INY
9A48 D0 F6              BNE  PRINT     ;BACK TO LOOP
9A4A C8        INPDO    INY            ;SPACE A CHARACTER
9A4B A2 00              LDX  #0        ;POINTER TO BUFFER
9A4D A9 A0     GET      LDA  #160
9A4F 99 C0 8F           STA  CRT+960,Y  ;CURSOR
9A52 84 FB              STY  AD
9A54 86 FC              STX  AD+1      ;ENSURE .Y & .X ARE PRESERVED
9A56 20 E4 FF  GETWT    JSR  GETIN
9A59 F0 FB              BEQ  GETWT     ;WAIT ON INPUT
9A5B A4 FB              LDY  AD
9A5D A6 FC              LDX  AD+1      ;BRING BACK .X,.Y
9A5F C9 0D              CMP  #13
9A61 D0 07              BNE  DELCK     ;NOT A RETURN
9A63 A9 20              LDA  #32
9A65 99 C0 8F           STA  CRT+960,Y  ;KILL CURSOR
9A68 8A                 TXA            ;SET ZERO FLAG AND LEAVE IN .A
9A69 60                 RTS            ;END OF INPUT
9A6A C9 14     DELCK    CMP  #20
9A6C D0 0E              BNE  RNGCK     ;NOT A DELETE
9A6E E0 00              CPX  #0
9A70 F0 DB              BEQ  GET       ;NOTHING IN BUFFER
9A72 A9 20              LDA  #32
9A74 99 C0 8F           STA  CRT+960,Y  ;CLEAR CURSOR
9A77 88                 DEY            ;BACKSPACE ON SCREEN
9A78 CA                 DEX            ;BACK UP IN BUFFER
9A79 4C 4D 9A           JMP  GET       ;BACK TO INPUT
9A7C C9 20     RNGCK    CMP  #32
9A7E 90 CD              BCC  GET       ;CONTROL CODES NOT ACCEPTED
9A80 C9 60              CMP  #96
9A82 B0 C9              BCS  GET       ;NOR GRAPHICS CODES
9A84 C0 27              CPY  #39
9A86 F0 C5              BEQ  GET       ;AT END OF LINE
```

```
9A88 9D 00 02          STA  INBUFF,X   ;IN LINE BUFFER
9A8B 29 3F             AND  #63        ;MODIFY FOR POKING
9A8D 99 C0 8F          STA  CRT+960,Y  ;PUT ON SCREEN
9A90 C8                INY             ;MOVE TO NEXT SPACE
9A91 E8                INX
9A92 4C 4D 9A          JMP  GET
                ;
9A95 20 C0 FF XFERIT   JSR  OPEN       ;OPEN THE FILE
9A98 B0 10             BCS  XFSTOP     ;CARRY SET = ERROR
9A9A 20 B7 FF          JSR  READST
9A9D D0 0B             BNE  XFSTOP     ;ZERO CLEAR = ERROR
9A9F A2 01             LDX  #1         ;LOGICAL FILE
9AA1 A5 02             LDA  LDSV
9AA3 D0 06             BNE  OUTPT      ;SAVING
9AA5 20 C6 FF          JSR  CHKIN      ;SET UP AS INPUT
9AA8 90 06             BCC  START      ;IF NO ERROR

9AAA 60       XFSTOP   RTS             ;ERROR - RETURN
9AAB 20 C9 FF OUTPT    JSR  CHKOUT     ;SET UP AS OUTPUT
9AAE B0 FA             BCS  XFSTOP     ;IF AN ERROR
9AB0 20 B7 FF START    JSR  READST
9AB3 D0 F5             BNE  XFSTOP     ;IF AN ERROR
9AB5 A9 D0             LDA  #>VIC
9AB7 85 FC             STA  AD+1       ;SET I/O TO BORDER COLOR
9AB9 A9 20             LDA  #32
9ABB 85 FB             STA  AD
9ABD 20 5D 9B          JSR  IO         ;SEND OR RECEIVE FROM/TO 53280
9AC0 E6 FB             INC  AD         ;NOW AT 53281
9AC2 20 5D 9B          JSR  IO         ;SEND/RECEIVE

                ;
9AC5 A9 00             LDA  #<CRT
9AC7 85 FB             STA  AD         ;SET UP I/O AT START OF SCREEN
9AC9 A9 8C             LDA  #>CRT
9ACB 85 FC             STA  AD+1       ;HIGH BYTE
9ACD 20 5D 9B TLP1     JSR  IO
9AD0 E6 FB             INC  AD
9AD2 D0 F9             BNE  TLP1       ;WAIT TILL END OF PAGE
9AD4 E6 FC             INC  AD+1       ;NEXT PAGE
9AD6 A5 FC             LDA  AD+1
9AD8 C9 8F             CMP  #>3*256+CRT
9ADA D0 F1             BNE  TLP1       ;NOT FINISHED
9ADC 20 5D 9B TLP2     JSR  IO         ;TRANSMIT/RECEIVE
9ADF E6 FB             INC  AD         ;NEXT BYTE
9AE1 A5 FB             LDA  AD
9AE3 C9 E8             CMP  #232
9AE5 D0 F5             BNE  TLP2       ;I/O 232 BYTES
                ;
9AE7 A9 00             LDA  #<COLCRT
9AE9 85 FB             STA  AD         ;SET FOR COLOR SCREEN
9AEB A9 D8             LDA  #>COLCRT
9AED 85 FC             STA  AD+1
9AEF 20 5D 9B CLP1     JSR  IO

9AF2 E6 FB             INC  AD
9AF4 D0 F9             BNE  CLP1
9AF6 E6 FC             INC  AD+1
9AF8 A5 FC             LDA  AD+1
9AFA C9 DB             CMP  #>3*256+COLCRT
9AFC D0 F1             BNE  CLP1       ;NOT FINISHED
9AFE 20 5D 9B CLP2     JSR  IO
9B01 E6 FB             INC  AD
9B03 A5 FB             LDA  AD
9B05 C9 E8             CMP  #232
9B07 D0 F5             BNE  CLP2       ;I/O LAST 232
                ;
9B09 A9 00             LDA  #<HIPAGE
9B0B 85 FB             STA  AD         ;HIRES PAGE
9B0D A9 A0             LDA  #>HIPAGE
```

```
9B0F 85 FC              STA  AD+1
9B11 20 5D 9B HLP1      JSR  IO
9B14 A0 00              LDY  #0
9B16 B1 FB              LDA  (AD),Y
9B18 D0 3D              BNE  HLPDO     ;IF NON-ZERO
                   ;
                   ;EXECUTE NON-STANDARD ZERO-FLAGGED DATA
9B1A A5 02     ZERDO    LDA  LDSV
9B1C D0 22              BNE  OUTZR     ;IF SAVING
9B1E 20 CF FF           JSR  CHRIN
9B21 85 FD              STA  MISC      ;READ ADDRESS OF FIRST
9B23 20 CF FF           JSR  CHRIN
9B26 85 FE              STA  MISC+1    ;NON-ZERO BYTE FOLLOWING
9B28 A9 00     NXTZ     LDA  #0
9B2A A8                 TAY

9B2B 91 FB              STA  (AD),Y    ;ZERO THE ADDRESS
9B2D 20 92 9B           JSR  NEXTY     ;NEXT ADDRESS
9B30 B0 2A              BCS  CLS       ;CARRY SET = END OF I/O
9B32 A5 FB              LDA  AD
9B34 C5 FD              CMP  MISC
9B36 D0 F0              BNE  NXTZ      ;NOT TO END OF ZEROS
9B38 A5 FC              LDA  AD+1
9B3A C5 FE              CMP  MISC+1
9B3C D0 EA              BNE  NXTZ      ;NOT TO END OF ZEROS
9B3E F0 D1              BEQ  HLP1      ;RETURN TO STANDARD I/O
9B40 20 92 9B OUTZR     JSR  NEXTY     ;NEXT ADDRESS

9B43 90 06              BCC  ZECHEK    ;NOT AT END OF I/O
9B45 20 87 9B           JSR  SENDZE    ;SEND FINAL ADDRESS
9B48 4C 5C 9B           JMP  CLS       ;END OF SUBROUTINE
9B4B A0 00     ZECHEK   LDY  #0
9B4D B1 FB              LDA  (AD),Y
9B4F F0 EF              BEQ  OUTZR     ;STILL A ZERO BYTE
9B51 20 87 9B           JSR  SENDZE    ;SEND ADDRESS
9B54 20 5D 9B           JSR  IO        ;SEND THE NON-ZERO BYTE
                   ;RESUME STANDARD I/O
9B57 20 92 9B HLPDO     JSR  NEXTY
9B5A 90 B5              BCC  HLP1      ;NOT AT END YET
9B5C 60        CLS      RTS            ;FINISHED
                   ;
9B5D A5 02     IO       LDA  LDSV
9B5F D0 15              BNE  OUTPUT    ;IF SAVING
9B61 20 CF FF           JSR  CHRIN     ;GET THE BYTE
9B64 48                 PHA            ;PUSH ON STACK
9B65 B0 1C              BCS  STP2      ;ABORT IF AN ERROR
9B67 20 B7 FF           JSR  READST
9B6A F0 04              BEQ  ENCKST    ;NO ERROR
9B6C C9 40              CMP  #64
9B6E D0 13              BNE  STP2      ;A NON-EOT CONDITION
9B70 68        ENCKST   PLA            ;RECALL THE CHRIN
9B71 A0 00              LDY  #0        ;ZERO .Y FOR INDIRECT
9B73 91 FB              STA  (AD),Y    ;STORE ON SCREEN
9B75 60                 RTS            ;END OF I/O SUBROUTINE
9B76 A0 00     OUTPUT   LDY  #0
9B78 B1 FB              LDA  (AD),Y    ;GET A BYTE FROM SCREEN
9B7A 20 D2 FF           JSR  CHROUT    ;SEND IT OUT
9B7D 20 B7 FF           JSR  READST
9B80 D0 02              BNE  STP       ;IF ERROR

9B82 60                 RTS
9B83 68        STP2     PLA            ;PULL EXTRA PUSH
9B84 68        STP      PLA
9B85 68                 PLA            ;REMOVE A LEVEL OF SUBROUTINE
9B86 60                 RTS            ;EXIT DIRECTLY TO MAIN ROUTINE
9B87 A5 FB     SENDZE   LDA  AD
9B89 20 D2 FF           JSR  CHROUT    ;SEND LOW BYTE
9B8C A5 FC              LDA  AD+1
9B8E 20 D2 FF           JSR  CHROUT    ;SEND HIGH BYTE
```

```
9B91 60                 RTS
9B92 E6 FB     NEXTY    INC  AD
9B94 D0 04              BNE  ENDY      ;NO PAGE
9B96 E6 FC              INC  AD+1      ;NEXT PAGE
9B98 18                 CLC            ;CLEAR CARRY = NO END OF PAGE
9B99 60                 RTS            ;BACK TO XFERIT SUBROUTINE
9B9A A5 FC     ENDY     LDA  AD+1
9B9C C9 BF              CMP  #>31*256+HIPAGE  ;END OF HIRES PAGE
9B9E 90 04              BCC  RETY      ;NOT AT END
9BA0 A5 FB              LDA  AD
9BA2 C9 41              CMP  #65       ;CARRY SET IF PAST 64
9BA4 60        RETY     RTS
               ;
               ;DATA FOR CHARACTER CHECKING
               ;
9BA5 12 17 0A DIRKEY    .BYT 18,23,10,9,20,12,62,14
9BAD 08 02 04 DIRECT    .BYT %1000,%0010,%0100,%0001
9BB1 0A 06 05           .BYT %1010,%0110,%0101,%1001
               ;
9BB5 90 05 1C COLORS    .BYT 144,5,28,159,156,30,31,158
9BBD 81 95 96           .BYT 129,149,150,151,152,153,154,155
               ;
9BC5 88 87 86 FNCTNS    .BYT 136,135,134,133
               ;
9BC9 48 49 52 QUESTN    .ASC "HIRES SKETCHPAD - BY CHRIS METCALF"
9BEB 0D                 .BYT 13
9BEC 4D 55 4C           .ASC "MULTICOLOR MODE? N"
9BFE 9D 00              .BYT 157,0
               ;
9C00 10 0C 0F STLINE    .BYT 16,12,15,20,58,6,32,58,32,32,32,32
9C0C 20 0C 09           .BYT 32,12,9,14,5,32,4,18,1,23,58,32,32,32,32,3
9C1C 20 06 09           .BYT 32,6,9,12,12,20,15,58,32,32,32,32
9C28 37 35 33 FNUM      .ASC "7531"
9C2C 0F 06 06 DRMD      .BYT 15,6,6,32,15,14,32,32
9C34 0F 06 06 LNMD      .BYT 15,6,6,32,6,18,15,13,20,15,32,32,12,9,14,5
9C44 13 01 0D FLMD      .BYT 19,1,13,5,1,14,25,32



               ;
               ;THESE ARE THE SPRITE MATRICES
               ;
               ;SPRITE FOR HIRES MODE
9C4C 00 38 00 SPRITE    .BYT %00000000,%00111000,%00000000
9C4F 00 44 00           .BYT %00000000,%01000100,%00000000
9C52 00 44 00           .BYT %00000000,%01000100,%00000000
9C55 06 FE C0           .BYT %00000110,%11111110,%11000000
9C58 09 01 20           .BYT %00001001,%00000001,%00100000
9C5B 06 00 C0           .BYT %00000110,%00000000,%11000000
9C5E 04 00 40           .BYT %00000100,%00000000,%01000000
9C61 04 00 40           .BYT %00000100,%00000000,%01000000
9C64 04 00 40           .BYT %00000100,%00000000,%01000000
9C67 06 00 C0           .BYT %00000110,%00000000,%11000000
9C6A 09 01 20           .BYT %00001001,%00000001,%00100000
9C6D 06 FE C0           .BYT %00000110,%11111110,%11000000
9C70 00 38 00           .BYT %00000000,%00111000,%00000000
9C73 00 0C              .BYT %00000000,%00001100


               ;
               ;THE SPRITE FOR MULTICOLOR MODE
9C75 00 18 00           .BYT %00000000,%00011000,%00000000
9C78 00 24 00           .BYT %00000000,%00100100,%00000000
9C7B 00 24 00           .BYT %00000000,%00100100,%00000000
9C7E 03 7E C0           .BYT %00000011,%01111110,%11000000
9C81 04 81 20           .BYT %00000100,%10000001,%00100000
9C84 03 00 C0           .BYT %00000011,%00000000,%11000000
9C87 02 00 40           .BYT %00000010,%00000000,%01000000
```

```
9C8A 02 00 40          .BYT %00000010,%00000000,%01000000
9C8D 02 00 40          .BYT %00000010,%00000000,%01000000
9C90 03 00 C0          .BYT %00000011,%00000000,%11000000
9C93 04 81 20          .BYT %00000100,%10000001,%00100000
9C96 03 7E C0          .BYT %00000011,%01111110,%11000000
9C99 00 18 00          .BYT %00000000,%00011000,%00000000
9C9C 00 0C 00          .BYT %00000000,%00001100,0
```

SECONDA PARTE

Come ridefinire il set dei caratteri

Capitolo uno

Costruite i vostri caratteri personalizzati

Orson Scott Card

Con parecchi programmi dimostrativi viene chiaramente spiegato l'uso dei caratteri personalizzati per tracciare disegni.

Il tipo di grafica di uso più semplice con il Commodore 64 è quello che utilizza i caratteri grafici. Dopo tutto usate caratteri grafici ogni volta che premete un tasto. Esiste un set di caratteri predefinito che stabilisce quale lettera debba apparire in una particolare posizione dello schermo quando premete un determinato tasto, ma l'aspetto grafico dell'operazione, l'effettiva comparsa del carattere sullo schermo, è in realtà piuttosto semplice.

Ecco come funziona.

Il VIC-II, il chip video del vostro calcolatore, dice al televisore che cosa visualizzare. Fino a che il VIC-II funziona lo schermo non è mai vuoto, in realtà. Sessanta volte al secondo il VIC-II dice al televisore che cosa porre in ogni singola posizione dello schermo. Visualizzare uno schermo vuoto non è più rapido né più facile che mostrarne uno con molte scritte; per il chip VIC-II è esattamente la stessa cosa. Un'area vuota dello schermo è piena di caratteri, esattamente come un'area che contenga dei testi. La differenza consiste nel fatto che l'area vuota è piena di caratteri «spazio vuoto», il carattere che ottenete premendo la barra spaziatrice.

Quindi lo schermo è sempre pieno di caratteri: tutto ciò che fate consiste nel cambiare il carattere che viene mostrato in un particolare punto.

Come parlare al televisore...

Non lo fate, tuttavia, mandando messaggi al televisore. Il televisore non ha memoria; potete segnalargli di porre un punto sullo

schermo in una determinata posizione, e lo potrebbe fare, ma esattamente 60 secondi più tardi il televisore cancella il punto, a meno che gli diciate di rimetterlo. Dovete inviare al televisore una nuova istruzione per ogni punto dello schermo, 60 volte al secondo. Se dovreste programmare tutto ciò per vostro conto, persino in linguaggio macchina, non vi rimarebbe molto tempo per altro. Ed in BASIC non potreste fare altro, in modo assoluto.

Fortunatamente, il chip VIC-II sa come parlare il linguaggio del televisore, altrettanto velocemente dello stesso. Tutto ciò che dovete fare è dire al VIC-II che cosa volete ed egli automaticamente dirà al televisore di farlo. Particolarmente importante è il fatto che *continuerà* a dire al televisore le stesse cose fino a che voi le cambierete o spegnerete il televisore.

La memoria di schermo

Il VIC-II comprende parecchie istruzioni differenti, ma non impartite queste istruzioni al VIC-II direttamente. Non dovete nemmeno creare un programma per VIC-II e mandarlo in esecuzione. Il VIC-II è già in esecuzione, fin dall'istante in cui avete acceso il calcolatore (c'è più di un modo per interrompere il VIC-II mentre il calcolatore è acceso, ma questo è un altro discorso).

Che cosa fa il VIC-II? Scandisce la memoria, ripetutamente, cercando specifiche istruzioni e dicendo al televisore che cosa fare. Ma non scandisce *tutta* la memoria. Cerca in determinate locazioni per trovare determinate cose. Dopo tutto, l'unica cosa che qualsiasi locazione può contenere è un valore numerico da 0 a 255. Lo stesso numero può significare cose diverse, in base a dove il VIC-II lo trova.

Per utilizzare i caratteri grafici, contrariarmente alla grafica ad alta risoluzione ed all'uso degli sprite, dovete solo conoscere alcune delle aree scandite dal VIC-II.

La memoria di schermo. La memoria di schermo ha una lunghezza di mille byte. Ciascun byte rappresenta una piccola area dello schermo. La memoria di schermo, come tutta la memoria dei calcolatori, è composta da una lunga fila di locazioni di memoria. Ma il VIC-II la legge come se si trattasse delle pagine di un libro.

Il VIC-II scandisce la memoria di schermo da 0 a 999. Il byte 0 rappresenta l'angolo superiore sinistro dello schermo, proprio come voi avete iniziato a leggere questa pagina partendo dall'angolo superiore sinistro.

Il byte che il VIC-II trova in quella posizione è un numero di codice simbolico che identifica un carattere. Questo non è il codice carattere ASCII, tuttavia. Questo è il *codice schermo.* Il VIC-II dice al televisore di visualizzare nell'angolo superiore sinistro qualsiasi carattere che viene richiesto. (L'appendice G contiene una tavola completa di tutti i codici schermo riconosciuti dal VIC-II).

Dopo aver riconosciuto il primo codice di schermo il VIC-II legge la successiva locazione di memoria, il byte 1. Il carattere richiesto verrà visualizzato immediatamente alla destra del carattere 0. Il byte 2 controlla il carattere successivo verso destra e così via. Dopo il byte 39 il VIC-II scende alla posizione all'estrema sinistra della seconda riga di schermo, proprio come fanno i vostri occhi quando leggono l'ultima parola di una riga. I byte da 0 a 39 rappresentano la prima riga di schermo; i byte da 40 a 79 la seconda riga; i byte da 80 a 119 la terza riga e così via, fino ai byte da 960 a 999 che completano l'ultima riga.

Quindi il VIC-II ha terminato la scansione della memoria di schermo, ma meno di 1/60 di secondo più tardi ritorna a scandirla da capo. Se avete cambiato il codice in un qualsiasi punto dello schermo dall'ultima volta che il VIC-II ha letto quella locazione, esso leggerà il nuovo valore ed inserirà un carattere differente sullo schermo in quella locazione. Il VIC-II non si preoccupa di quale codice trova in una qualsiasi locazione, né succede niente di particolare se avete cambiato carattere; il VIC-II si limita a prendere quello che trova ed a passarlo al televisore.

La memoria di schermo, quindi, diventa una mappa dello schermo. Utilizzando una istruzione PEEK alla memoria di schermo il vostro programma può scoprire che caratteri vengono attualmente mostrati sul televisore; con una istruzione POKE alla memoria di schermo il vostro programma è in grado di cambiare ciò che il televisore mostra.

La memoria carattere. Il codice schermo di per sé non è sufficiente per porre un carattere completo sullo schermo. Il codice schermo è solo un *indice* di ricerca all'interno del set dei caratteri.

Come la memoria di schermo, la memoria carattere è solo una sezione di memoria. La forma di ciascun carattere è registrata in una serie di otto byte. Dal momento che il Commodore 64 ha accesso a 512 caratteri differenti (due serie di 128 caratteri e di 128 caratteri in negativo ciascuna), la memoria carattere consiste di otto volte

altrettanti byte: 4096 byte, per essere esatti. Cioè 4Kbyte, molto più grande della memoria di schermo.

Le forme dei caratteri composte da otto byte sono registrate nello stesso ordine del codice schermo. Il primo codice schermo è lo 0, che equivale al carattere @. Il secondo codice schermo, 1, è la A; i successivi otto byte nella memoria carattere contengono la forma della lettera A.

Questo significa che per trovare la forma di qualsiasi carattere in memoria carattere tutto quello che dovete fare consiste nel prendere il codice schermo del carattere e moltiplicarlo per otto, quindi contare altrettanti byte dall'inizio della memoria carattere. Il codice schermo per la Z è 26. Otto per 26 è uguale a 208. Quindi il primo byte della figura che compone la lettera Z è il 208-esimo della memoria carattere.

Ogni volta che il VIC-II legge un numero di codice schermo conta il numero opportuno di byte dall'inizio della memoria carattere per cercare la forma del carattere voluto. Quindi dice al televisore di mostrarne la forma.

Come potete crearvi dei caratteri personalizzati? Cambiando le forme registrate nella memoria carattere. Il VIC-II non può leggere l'alfabeto. Non si preoccupa se la forma del carattere corrispondente al codice schermo 1 sembra una A o no. Stamperà qualsiasi figura si trovi in quella sezione di otto bit della memoria carattere, senza far domande. Potrebbe essere una lettera dell'alfabeto cirillico o il disegno di un papero; una volta che la forma contenuta nel set di caratteri è stata cambiata, è quella che verrà mostrata ogni volta che il VIC-II trova quel particolare codice schermo.

La memoria colore. Oltre alla memoria di schermo, che rappresenta una mappa dei caratteri che si trovano sullo schermo, esiste una seconda mappa che tiene traccia dei colori sullo schermo. Potete scegliere il colore per ciascun carattere, individualmente, nella memoria di schermo, cambiando la corrispondente locazione della memoria colore.

La mappa dei caratteri dei codici schermo e la mappa colore dei codici colore hanno una corrispondenza esattamente biunivoca. Vale a dire, qualsiasi carattere venga richiamato nel byte 299 della memoria di schermo verrà visualizzato con qualsiasi colore venga richiesto dal byte 299 della memoria colore.

Capitolo uno

Come si sposta la memoria carattere

Quando accendete il vostro calcolatore la memoria di schermo ha inizio in 1024, la memoria colore in 55296 e la memoria carattere in 53248. Ma ciò non è necessariamente fissato in modo permanente. Potete dire al VIC-II di cercare la memoria di schermo, la memoria colore e la memoria carattere in qualche altro punto. Per i nostri scopi attuali non è necessario spostare la memoria di schermo o quella colore. *Dovremo*, invece, spostare la memoria carattere.

Perché dobbiamo spostarla? Perché non possiamo inserire la nuova forma dei caratteri, con delle istruzioni POKE, nella memoria carattere nella sua posizione attuale?

La ragione è abbastanza semplice. Il set di caratteri risiede in ROM, (Read Only Memory, memoria a sola lettura, inalterabile). Fintanto che il VIC-II cerca la forma dei caratteri in ROM non potete cambiarne la forma. Ecco perché il set dei caratteri non viene cancellato ogni volta che spegnete il vostro calcolatore.

Quindi, prima che il VIC-II possa cominciare ad usare il vostro nuovo set di caratteri dovete segnalargli di cercarlo altrove. Potete farlo cambiando il valore numerico registrato nella locazione 53272.

Dove mettere la memoria carattere. La tentazione di addentrarsi nei meandri dell'organizzazione della memoria dedicata alla grafica è molto forte, ma non è necessario farlo per gli scopi che ci siamo prefissi. È sufficiente dire che il chip VIC-II può «vedere» solo un blocco di 16Kbyte della RAM per volta e che, quindi, la memoria di schermo ed il set di caratteri devono sempre essere compresi nello stesso blocco, a meno che usiate il set di caratteri incorporato.

Dal momento che la memoria di schermo e la memoria carattere devono stare nello stesso blocco, il VIC-II cerca solo delle istruzioni che gli dicano *quale* blocco di memoria da 2Kbyte, all'interno del blocco da 16Kbyte, contiene il set dei caratteri. Quindi esistono solo otto possibili informazioni di allocazione della memoria carattere. I numeri di codice dei blocchi sono i numeri pari da 0 a 14. Questi numeri di codice, registrati nella locazione 53272, dicono al VIC-II di cercare la memoria carattere in una delle otto possibili locazioni all'interno del blocco:

istruzione	**indirizzo di partenza all'interno del blocco**
0	0

2	2048
4	4096
6	6144
8	8192
10	10240
12	12288
14	12336

(Per una trattazione più approfondita dei blocchi della memoria dedicata alla grafica vedi «La memoria dedicata alla grafica» nel Capitolo 2 del Volume 1°. Per ora assumiamo semplicemente di usare il blocco grafico per difetto, quello che inizia alla locazione 0 e va fino alla 16383).

Le informazioni riguardanti la memoria carattere. Perché sono necessari solo 2Kbyte per contenere la memoria carattere, quando il set di caratteri ROM è lungo 4096 byte? Ciò avviene perché il set di caratteri ROM è in realtà *due* set di caratteri. Il VIC-II li legge entrambi, ma solo uno per volta. Ciascuno è lungo 2Kbyte. Potete creare fino a sette set di caratteri contemporaneamente, ed alternarli cambiando semplicemente il contenuto della locazione 53272. È la stessa cosa che succede quando premete SHIFT ed il tasto Commodore contemporaneamente: state semplicemente alternando i set di caratteri.

Cambiare il codice in 53272 non è, tuttavia, così semplice come inserire un numero di codice con una POKE. Ciò a causa del fatto che la locazione della memoria carattere è registrata nei bit 1-3. I bit 4-7 sono utilizzati per registrare la locazione della memoria di schermo. Se inseriste, con una istruzione POKE, l'informazione di memoria carattere 12 nella locazione 53272, il numero binario ivi registrato sarebbe:

	bit:	7	6	5	4	3	2	1	0
12		0	0	0	0	1	1	0	0
						codice			

Notate che i bit 4-7, che contengono le informazioni riguardanti la memoria di schermo, contengono solo zeri. Quindi il VIC-II cercherà la memoria di schermo nella locazione 0 all'interno del blocco. Dal momento che questa sezione di memoria viene utiliz-

zata per funzioni vitali del linguaggio macchina, il vostro schermo avrà un aspetto strano.

Fortunatamente, il BASIC del Commodore 64 comprende due operatori che permettono di cambiare singoli bit in un byte senza alterare il resto del byte: gli operatori AND e OR (se non li sapete già usare, potete vedere «Impariamo la grafica «bitmap» nel Capitolo 2 del Volume 1°. Ecco come cambiare la locazione contenente il set di caratteri nel codice 12, senza cambiare le informazioni riguardanti la memoria di schermo:

POKE 53272,(PEEK(53272) AND 240) OR 12

Per specificare una diversa locazione di memoria carattere cambiate il valore numerico 12 in un diverso numero pari, da 0 a 14.

I set di caratteri misti. Cambiare semplicemente la locazione in cui il VIC cerca il set di caratteri non vi pone effettivamente i caratteri stessi, ovviamente. L'unico modo di porre la forma dei caratteri nella memoria carattere, una volta che avete rinunciato ad usare il set di caratteri ROM, consiste nel mettere da voi stessi la loro forma nel punto voluto.

Spesso desidererete mescolare i set di caratteri. Vale a dire, vorrete usare contemporaneamente le lettere dell'alfabeto ed alcuni simboli del set proveniente dalla ROM unitamente ad alcuni dei vostri caratteri personalizzati. Il modo più semplice per farlo consiste nel copiare il set di caratteri della ROM — o parte di esso — nella nuova area di memoria carattere. Una volta che è al suo posto, cambiate solo la forma di un numero limitato di caratteri, quelli che volete personalizzare. Gli altri conserveranno l'aspetto del set ROM.

Per copiare il set della ROM dovete utilizzare innanzi tutto un paio di istruzioni POKE. Non è necessario che comprendiate l'aspetto ingegneristico della faccenda. La ROM carattere è un banco di memoria normalmente escluso dall'accesso, dove non potete operare tramite istruzioni PEEK. Dovete collegarvi ad esso, ma prima di farlo dovete disabilitare tutte le interruzioni. Quindi, quando avete terminato di copiare il set di caratteri della ROM dovete togliere il collegamento e riabilitare le interruzioni. Ecco la serie di istruzioni che svolge questo compito:

disabilitazione delle interruzioni:

POKE 56334, PEEK (56334) AND 254

collegamento con la ROM carattere:

POKE 1, PEEK(1) AND 251

(a questo punto potete copiare il set di caratteri ROM).
disinserimento del collegamento con la ROM carattere:

POKE 1, PEEK(1) OR 4

abilitazione delle interruzioni:

POKE 56334, PEEK(56334) OR 1

Ora possiamo compiere la nostra prima semplice operazione sul set di caratteri. Il seguente programma copierà il set di caratteri ROM e dirà al VIC-II che può trovare la memoria carattere alla locazione di codice 12.

```
10 CM=12288:CX=53248
15 GOSUB800
20 POKE53272,(PEEK(53272)AND240)OR12
199 END
800 POKE56334,PEEK(56334)AND254
805 POKE1,PEEK(1)AND251
810 FORI=1TO1023:POKECM+I,PEEK(CX+I):NEXT
815 POKE1,PEEK(1)OR4
820 POKE56334,PEEK(56334)OR1
825 RETURN
```

Il problema che affligge questo programma è rappresentato dal fatto che non fa nulla che voi possiate vedere. Si limita a copiare la ROM, per cui, per quel che potete notare, il computer ha la stessa configurazione che ha sempre avuto. Ora è il momento di cominciare a cambiare l'aspetto dei caratteri.

La forma dei caratteri

La forma di ciascun carattere consiste di otto byte. Ciascun byte consiste di otto bit. Ciò significa che ciascun carattere può essere rappresentato in questo modo:

bit 7 6 5 4 3 2 1 0

byte
0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0
3 0 0 0 0 0 0 0 0
4 0 0 0 0 0 0 0 0
5 0 0 0 0 0 0 0 0
6 0 0 0 0 0 0 0 0
7 0 0 0 0 0 0 0 0

Lo schermo televisivo è suddiviso in 25 righe di 40 caratteri ciascuna. Ogni carattere consiste di un quadratino di otto punti per lato. Questo quadratino ha una corrispondenza biunivoca con i bit degli otto byte che costituiscono la figura dei caratteri. Ciascun bit è uguale a 1 o a 0. Se il bit è uguale a 0, il punto corrispondente dello schermo è *spento* e viene visualizzato nel colore di fondo. Se il bit è a 1, allora il punto è *acceso* e viene visualizzato il colore di primo piano.

La Figura 1 mostra la forma che genera la lettera A. Ciascun bit *attivato*, o uguale a 1, sarà acceso sullo schermo; ciascun bit *disattivato*, o uguale a 0, sarà spento. Una volta tolti gli zero potete facilmente vedere la forma che verrà effettivamente visualizzata.

Alla destra di ciascun byte nella figura c'è un numero decimale. Questo è il numero che dovreste inserire, con una istruzione POKE, nella posizione corrispondente della figura del carattere, in modo da ottenere quel byte, con quella serie di bit *accesi* e *spenti.*

Per mostrare come potete cambiare queste forme, ecco un programma che aggiunge una linea retta attraverso ciascuno dei primi otto caratteri del set: @, A, B, C, D, E, F e G. È identico al primo programma, eccettuata la riga 25. Per vedere qual è l'aspetto delle nuove lettere limitatevi a batterle non appena il programma è terminato. Il messaggio READY mostrerà già che cosa è accaduto alle lettere A e D.

```
10 CM=12288:CX=53248
15 GOSUB800
20 POKE53272,(PEEK(53272)AND240)OR12
25 FORI=0TO63STEP9:POKECM+I,255:NEXT
199 END
800 POKE56334,PEEK(56334)AND254
```

```
805 POKE1,PEEK(1)AND251
810 FORI=1TO1023:POKECM+I,PEEK(CX+I):NEXT
815 POKE1,PEEK(1)OR4
820 POKE56334,PEEK(56334)OR1
825 RETURN
```

Che cosa fa effettivamente la riga 25? Innanzi tutto ricordatevi che il numero 255 è il più alto numero rappresentabile di otto bit. Tutti i bit che lo compongono sono uguali a 1. Quindi, se un byte nella forma del carattere ha valore numerico decimale 255, verrà visualizzato come una sottile linea orizzontale.

Figura 1. Mappa carattere della lettera A.

byte	bit 7 6 5 4 3 2 1 0		decimale
0	0 0 0 0 0 0 0 0		0
1	0 0 0 1 1 0 0 0	11	24
2	0 0 1 1 1 1 0 0	1111	60
3	0 1 1 0 0 1 1 0	11 11	102
4	0 1 1 1 1 1 1 0	111111	126
5	0 1 1 0 0 1 1 0	11 11	102
6	0 1 1 0 0 1 1 0	11 11	102
7	0 0 0 0 0 0 0 0		0

Figura 2. Matrice per disegnare i caratteri.

Byte	128	64	32	16	8	4	2	1	valore decimale
0									______
1									______
2									______
3									______
4									______
5									______
6									______
7									______

La riga 25, quindi, inserirà con una POKE il valore numerico 255 una sola volta nelle forme dei primi otto caratteri. Dal momento che il passo (STEP) del ciclo FOR-NEXT è 9 e non 8, la linea retta apparirà in una posizione di un gradino inferiore in ciascuna figura rispetto al carattere precedente.

Come combinare le forme. È possibile anche combinare la forma dei caratteri. In questo programma le righe 25 e 30 prelevano la forma dei caratteri 73 e 74 (piccoli segmenti circolari) e li fondono in un'unica forma al posto del carattere @. La forma dei caratteri viene combinata applicando l'operatore OR ai byte 0 di entrambe le forme, ripetendo quindi l'operazione per i byte da 1 a 7. Per vedere combinazioni di caratteri differenti cambiate i valori X ed Y in riga 25. La maggior parte delle combinazioni, tuttavia, non sarà troppo chiara.

```
10 CM=12288:CX=53248
15 GOSUB800
20 POKE53272,(PEEK(53272)AND240)OR12
25 X=73:Y=74
30 FORI=0TO7:POKECM+I,PEEK(CM+I+8*X)ORPEEK
   (CM+I+8*Y):NEXT
199 END
800 POKE56334,PEEK(56334)AND254
805 POKE1,PEEK(1)AND251
810 FORI=1TO1023:POKECM+I,PEEK(CX+I):NEXT
815 POKE1,PEEK(1)OR4
820 POKE56334,PEEK(56334)OR1
825 RETURN
```

Animazioni. Disegnando parecchi caratteri solo leggermente differenti l'uno dall'altro e quindi inserendoli successivamente nella stessa locazione di memoria schermo, tramite istruzioni POKE, è possibile creare l'illusione del movimento. In questo programma le righe 20-30 creano una serie di otto caratteri. Tutti hanno una linea diagonale ed una verticale, ma in ciascun carattere la linea verticale si trova in una posizione leggermente differente. Questo programma crea solo i caratteri: aggiungeremo il movimento tra un attimo:

```
10 CM=12288:CX=53248
15 GOSUB800
20 POKE53272,(PEEK(53272)AND240)OR12
25 FORI=0TO7:FORJ=0TO7:W=I*8+J:POKECM+W,(2
   ↑(I+2))/4OR(2↑(J+2))/4
30 NEXT:NEXT
199 END
800 POKE56334,PEEK(56334)AND254
805 POKE1,PEEK(1)AND251
810 FORI=1TO1023:POKECM+I,PEEK(CX+I):NEXT
815 POKE1,PEEK(1)OR4
820 POKE56334,PEEK(56334)OR1
825 RETURN
```

Battendo i caratteri da @ a G da tastiera, potete vedere qual è l'aspetto dei nuovi caratteri.

Ora è venuto il momento di aggiungere il sottoprogramma di animazione. In questo programma viene cambiata la riga 10; SM viene inizializzata all'indirizzo di partenza della memoria di schermo e CL all'indirizzo iniziale della memoria colore. Il ciclo principale del programma, le righe da 100 a 130, inseriscono nella locazione 500 della memoria di schermo il colore bianco e successivamente i codici di schermo da 0 a 7, tramite una istruzione POKE. Quando questo ciclo è stato completato la riga 120 fa la stessa cosa, ma in senso inverso.

La riga 200 è un sottoprogramma di ritardo. Senza di esso il movimento sarebbe troppo rapido per poter essere visibile. Quando il programma è in esecuzione sembrerà che la linea verticale si muova avanti e indietro attraverso la linea diagonale.

```
10 CM=12288:CX=53248:SM=1024:CL=55296
15 GOSUB800
20 POKE53272,(PEEK(53272)AND240)OR12
25 FORI=0TO7:FORJ=0TO7:W=I*8+J:POKECM+W,(2
   ↑(I+2))/4OR(2↑(J+2))/4
30 NEXT:NEXT
100 POKECL+500,1
110 FORI=0TO7:POKESM+500,I:GOSUB200:NEXT
```

```
120 FORI=7TO0STEP-1:POKESM+500,I:GOSUB200:
    NEXT
130 GOTO110
199 END
200 FORJ=0TO50:NEXT:RETURN
800 POKE56334,PEEK(56334)AND254
805 POKE1,PEEK(1)AND251
810 FORI=1TO1023:POKECM+I,PEEK(CX+I):NEXT
815 POKE1,PEEK(1)OR4
820 POKE56334,PEEK(56334)OR1
825 RETURN
```

Disegnare con le istruzioni DATA. Finora tutti i nuovi caratteri sono stati disegnati usando espressioni matematiche. In pratica, questo metodo non viene quasi mai usato nella programmazione. Invece, traccerete la forma di ciascun carattere separatamente, e inserirete gli stessi caratteri usando istruzioni DATA. Il metodo più chiaro consiste nel mettere le otto istruzioni DATA per la forma di un particolare carattere su di una sola riga:

DATA 0,9,22,128,255,128,66,0

Notate che nel punto in cui volete ottenere una riga vuota dovete inserire uno 0. La forma di ciascun carattere deve avere otto byte; i byte vuoti devono essere rappresentati da 0.

Come potete tracciare i vostri caratteri e calcolare i valori decimali? Il modo più facile consiste nell'usare un programma dedicato alla creazione dei caratteri che vi permette di vedere la forma esatta del carattere che state creando, senza dover calcolare, per alcun carattere, la configurazione dei bit ed i valori decimali corrispondenti. (Un ottimo esempio di programma dedicato alla creazione dei caratteri è pubblicato nell'articolo seguente: «Un assortimento completo di caratteri da stampa»).

Tuttavia, per creare solo alcuni caratteri potete usare questo semplice metodo. Tracciate una griglia 8 per 8 (oppure segnate una sezione quadrata 8x8 su di un foglio normale di carta da lucido). Riempite ogni riquadro che volete attivare; lasciate in bianco ciascun riquadro che dovrebbe avere lo stesso colore di fondo. Quando avete ottenuto la forma voluta ciascuna riga orizzontale rappresenta un singolo byte della figura, disposto ordinatamente dall'alto

in basso. Ciascun riquadro campito rappresenta un bit *attivato*, o a 1. Ogni bit *attivato* avrà un valore differente, dipendente dalla sua posizione internamente alla riga. Un 1 nella posizione più a sinistra ha un valore di 128; un 1 nella posizione più a destra ha un valore unitario. Uno zero, cioè uno spazio vuoto, in qualsiasi posizione ha valore nullo.

Questa tavola mostra i valori. Per calcolare il valore decimale del numero binario 01110011 sommate i valori dei bit *attivati*. In questo caso i valori, da sinistra a destra, sono 64, 32, 16, 2 e 1. Quindi, il valore numerico decimale che produrrà questa disposizione di bit è uguale a 64+32+16+2+1, cioè 115. Questo è il valore numerico che dovreste inserire, tramite una POKE, nella memoria carattere per riprodurre questa disposizione di bit.

bit	7	6	5	4	3	2	1	0
		X	X	X			X	X
valore decimale dei bit *attivati*	128	64	32	16	8	4	2	1

Se questa è la riga superiore della disposizione che costituisce il carattere, mettete questo valore numerico come primo nell'istruzione DATA; se è l'ultima riga, lo dovrete inserire per ultimo. Quindi leggete con una istruzione READ le righe DATA internamente ad un ciclo ed inserite i valori nella memoria caratteri tramite istruzioni POKE. Se avete disposto le istruzioni DATA nell'ordine esatto, la forma dei caratteri, alla fine del ciclo, sarà quella voluta.

Questo programma costruisce un carattere molto semplice, che rimpiazza il carattere @. Avrà l'aspetto di quattro righe orizzontali. Queste hanno valore numerico 255; gli spazi tra di esse, naturalmente, 0.

```
10 CM=12288:CX=53248
15 GOSUB800
20 POKE53272,(PEEK(53272)AND240)OR12
```

```
25 FORI=0TO7:READN:POKECM+I,N:NEXT
199 END
800 POKE56334,PEEK(56334)AND254
805 POKE1,PEEK(1)AND251
810 FORI=1TO1023:POKECM+I,PEEK(CX+I):NEXT
815 POKE1,PEEK(1)OR4
820 POKE56334,PEEK(56334)OR1
825 RETURN
900 DATA255,0,255,0,255,0,255,0
```

Dopo che questo programma è stato mandato in esecuzione premete il tasto @ per vedere il nuovo carattere.

Animazioni con caratteri combinati. Questo programma mostra come sia possibile ottenere una animazione regolare, progettando attentamente i caratteri personalizzati. Le figure saranno sempre costituite da due caratteri, uno raffigurante la metà superiore della figura umana, l'altro la metà inferiore. Inserendo la metà superiore, tramite una POKE, nella locazione di memoria di schermo 500 e la metà inferiore in 540 (esattamente una riga al di sotto), si otterrà una figura umana completa.

Ci sono otto caratteri coinvolti nella sequenza delle animazioni, quattro per le metà superiori e quattro per le inferiori. Ciascun carattere è solo leggermente diverso dal precedente. Inserendoli in memoria nell'ordine opportuno, sembrerà che la figura corra sul posto. Dal momento che questo programma non utilizza nessuno dei caratteri del set ROM normale, non è stata inclusa la routine per copiare la ROM.

```
10 CM=12288:SM=1024:CL=55296
20 POKE53272,(PEEK(53272)AND240)OR12
25 FORI=0TO63:READN:POKECM+I,N:NEXT
100 POKECL+500,1:POKECL+540,1
110 FORI=0TO6STEP2:POKESM+500,I:POKESM+540
    ,I+1:GOSUB200:NEXT:GOTO110
130 GOTO110
199 END
200 FORJ=0TO99:NEXT:RETURN
900 DATA0,0,3,3,30,44,76,140
910 DATA12,10,17,18,20,18,32,16
920 DATA0,0,6,6,12,28,28,30
```

```
930 DATA12,10,18,17,34,51,130,64
940 DATA0,12,12,24,24,28,30,14
950 DATA12,10,10,10,18,34,66,1
960 DATA0,0,6,6,8,29,46,76
970 DATA12,12,12,66,72,8,8,12
```

Le Figure da 3 a 6 mostrano la forma degli otto caratteri usati per creare l'effetto di animazione, insieme ai valori numerici decimali che vengono effettivamente inseriti in memoria con l'istruzione POKE.

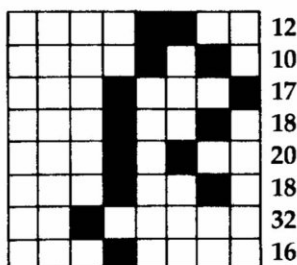
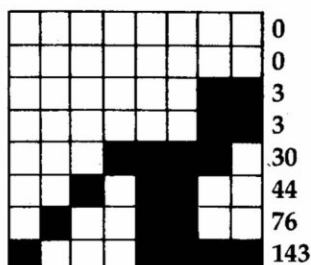
Figura 3.

Prima figura in corsa

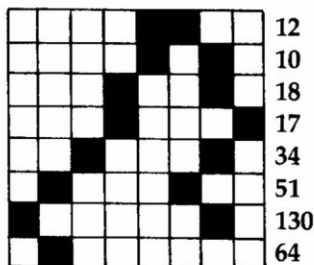
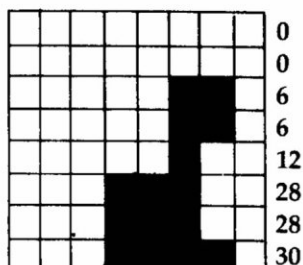
Figura 4.

Seconda figura in corsa

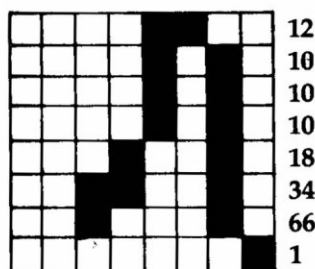
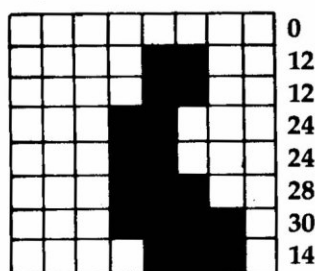
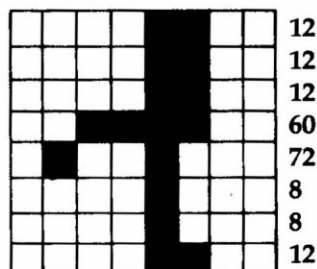
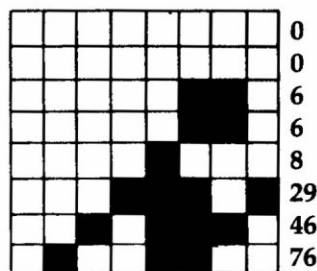
Figura 5.

Terza figura in corsa

Figura 6.

Quarta figura in corsa

Capitolo uno

Un assortimento completo di caratteri da stampa

Charles Brannon

Questo programma di utilità pratica, veloce, compatto, completamente in linguaggio macchina, facilita la progettazione di caratteri personalizzati.

Chiunque ha usato la carta da lucido per tracciare caratteri, e quindi ha affrontato il noioso compito di convertirli in numeri decimali, è in grado di apprezzare l'utilità di un programma dedicato alla costruzione di caratteri personalizzati. Invece di disegnare e cancellare sulla carta, potete tracciare i vostri caratteri liberamente con un joystick. «Un assortimento completo di caratteri da stampa» è stato scritto per offrire praticamente ogni possibile ausilio per aiutarvi a disegnare l'intero set di caratteri.

Come copiarlo

Questo programma, interamente scritto in linguaggio macchina, vi fornisce quella velocità ed efficienza che il BASIC non può mettervi a disposizione. Però, pur fornendovi un prodotto di elevata qualità, porta con sé l'inconveniente di un lungo lavoro di copiatura. Il programma è in effetti piuttosto corto, dal momento che usa meno di 3Kbyte di memoria a partire dalla locazione esadecimale $C000, riservata a programmi come questo. Quindi non perdete neppure un byte dello spazio di memoria programmabile in BASIC.

Tuttavia, 3000 caratteri richiedono un lavoro di copiatura almeno tre volte più lungo, dal momento che ciascun byte deve essere

rappresentato da un numero di tre cifre (000-255). Con un così lungo lavoro di copiatura gli errori sono inevitabili. Per facilitarvi il compito abbiamo predisposto l'«Assortimento completo di caratteri da stampa» perché possa essere copiato usando MLX, il programma per l'introduzione di programmi in linguaggio macchina. Istruzioni complete su quest'ultimo programma si possono reperire nell'articolo che lo accompagna (Appendice I). Quindi, nonostante le difficoltà comportate dal lavoro di copiatura, state pur certi che un paio di pomeriggi alla tastiera vi ricompenseranno interamente della fatica.

Una volta che avete copiato, registrato e mandato in esecuzione MLX, rispondete con 49152 e 52139, rispettivamente, alle due domande riguardanti l'indirizzo iniziale e quello finale. Dopo aver registrato il programma tramite MLX potete caricarlo da disco con LOAD «nome del file», 8, 1 o da nastro con LOAD «nome del file», 1, 1. Dopo averlo caricato battete NEW, quindi SYS 49152.Questo comando «manda in esecuzione» il programma in linguaggio macchina, che parte in $C000 (12*4096=49152).

Lo schermo

Dopo aver attivato il programma con una SYS verrà visualizzata l'area di lavoro. Ai piedi dello schermo troviamo otto righe di 32 caratteri. Sono i 256 caratteri che potete personalizzare. Un quadrato lampeggiante è posto sul simbolo @, la posizione di base del set di caratteri. Al di sopra delle otto righe sta la griglia principale, una vista ingrandita di dieci caratteri. L'ultima riga dello schermo è riservata ai messaggi.

A proposito della griglia

Questa griglia è come una finestra di grandi dimensioni aperta sul set di caratteri. Potete vedere i primi cinque caratteri ed i cinque caratteri sotto ad essi. Un grande cursore blu mostra su quale carattere state attualmente operando, ed un riquadro lampeggiante più piccolo è il cursore che usate per porre e rimuovere i punti.

Come ci si muove per lo schermo

Potete usare i tasti cursore (su, giù, a sinistra, a destra) per spostare il grande cursore blu su qualsiasi carattere che volete modificare. Se vi spostate su di un carattere che non compare nella

grande griglia (cioè se uscite dalla finestra), la finestra scorrerà automaticamente per far apparire il carattere. Potete anche guardare la parte inferiore dello schermo per spostare il grande cursore, dal momento che il quadratino lampeggiante sul set di caratteri si sposta con la griglia principale.

Il tasto HOME sposta il piccolo cursore nell'angolo superiore sinistro dello schermo. Se lo premete due volte di fila vi riporterà all'inizio del set di caratteri — a @.

Potete spostare il piccolo cursore all'interno della griglia usando un joystick inserito nella Port 2. Se muovete il cursore al di fuori del carattere corrente, il cursore blu salterà sul carattere adiacente in qualsiasi direzione vi vogliate spostare. Il display alla base dello schermo verrà aggiornato e la griglia scorrerà, se necessario. Ciò significa che potete ignorare i confini tradizionali tra ciascun carattere e tracciare forme larghe quanto l'intero set di caratteri (256x64 pixel: un pixel è l'unità elementare del disegno, un punto). Potete anche costruire un carattere per volta, oppure costruire una figura all'interno di un riquadro di due caratteri di lato.

Il pulsante di sparo viene utilizzato per porre e rimuovere punti. Se il cursore si trova su di un riquadro pieno, verrà spento. Se il riquadro è disattivato, verrà attivato. Se tenete premuto il pulsante di sparo mentre muovete il joystick, potete continuare nello stesso formato. Se tracciate un punto, continuerete a tracciarne mentre vi muovete. Se ne cancellate uno, potete muovervi cancellando punti lungo tutto lo schermo.

Se il cursore è troppo rapido o troppo lento per essere usato, premete V per regolare la velocità del cursore. Rispondete alla domanda con un valore di velocità da 0 (lento) a 9 (troppo veloce per essere usato).

Giochi di prestigio

Esistono parecchie funzioni che influenzano il carattere correte (quello su cui si trova il grande riquadro blu). Potete ruotare, far scorrere, riflettere, porre in negativo, cancellare, rimpiazzare e copiare caratteri. Il modo migliore di imparare ad usarle consiste nel giocarci. È veramente molto divertente! I tasti seguenti controllano ciascuna funzione:

Tasti di funzione:

f1: Fa scorrere i caratteri verso destra. Tutti i pixel si spostano a

destra. La colonna di pixel all'estrema destra ricompare all'estrema sinistra.

f2: Fa scorrere i caratteri verso sinistra. La ricomparsa dei pixel è analoga a quella provocata da f1.

f3: Fa scorrere i caratteri verso il basso. Tutti pixel si spostano verso il basso. L'ultima riga di pixel ricompare in cima al carattere.

f4: Fa scorrere i caratteri verso l'alto. La ricomparsa dei pixel è analoga a quella provocata da f3.

R: Rotazione. Fa ruotare i caratteri di 90 gradi. Premere due volte per capovolgere il carattere.

M: Riflessione. Crea una immagine riflessa della parte sinistra del carattere a destra.

CLR (SHIFT CLR/HOME): Cancella il carattere corrente.

CTRL-R o CTRL-9: Pone il carattere in negativo. Tutti i punti attivati vengono cancellati e tutti i punti disattivati vengono riempiti. La metà inferiore del set di caratteri è l'immagine in negativo della metà superiore.

F: Ripristina. Usate questa funzione se volete ripristinare la forma originale del carattere. Se avete ridefinito la A e premete F, verrà nuovamente copiato dalla ROM l'aspetto previsto dal calcolatore per la A.

Come registrare e caricare il set dei caratteri

Per registrare la vostra opera su nastro o su disco premete S. Quindi premete T per nastro (tape) o D per disco. Quando vi viene richiesto inserite il nome del file, fino a 16 caratteri. Non utilizzate il prefisso «0:», se usate l'unità disco. Lo schermo verrà cancellato, mostrerà i messaggi opportuni e quindi ritornerà al menu principale, se non ci sono errori. Se ci sono errori, come ad esempio mancanza di spazio sul disco, il programma leggerà il messaggio di errore dei dischi e lo mostrerà in fondo allo schermo. Premete un tasto dopo aver letto il messaggio e cercate di ovviare alle cause dell'errore, prima di registrare nuovamente il set di caratteri modificato. Il programma non può rilevare un errore di registrazione nel corso di una registrazione su nastro.

Per caricare un set di caratteri precedentemente registrato premete L e rispondete al messaggio «Tape or Disk». Segnalate il nome del file. Se utilizzate l'unità nastro, assicuratevi che il nastro sia riavvolto e pronto. Dopo il caricamento si ritornerà al menu iniziale; basta un'occhiata per verificare se il set di caratteri è stato caricato.

Capitolo uno

Se viene rilevato un errore durante il caricamento da nastro, vedrete apparire il messaggio «Error on Save/Load». Ancora, se usate i dischi, verrà mostrato il relativo messaggio di errore. Premete un tasto per ritornare al menu principale, in modo da riprovare.

Come copiare e spostare i caratteri

Potete copiare un carattere da un altro con i tasti funzione 7 e 8. Quando premete f7 il carattere corrente lampeggerà brevemente e verrà copiato in una piccola area tampone. Il programma ricorderà la forma di quel carattere. Potete quindi muovere il cursore dove volete copiare il carattere e premere SHIFT-f7 (f8). Il carattere memorizzato rimpiazzerà quindi il carattere su cui è posizionato il cursore. Potete anche utilizzare l'area tampone come dispositivo di sicurezza. Prima di cominciare a ridefinire un carattere su cui avete già lavorato premete f7 per registrarlo nell'area tampone, per maggior sicurezza. In questo modo, se accidentalmente lo cancellate o lo danneggiate, potete premere f8 per riacquisire il vostro carattere precedente.

Come si creano le istruzioni DATA

Un comando particolarmente utile, CTRL-D (tenete premuto CTRL e battete D), vi permette di creare istruzioni DATA per qualsiasi carattere che avete definito. Il programma non fornisce istruzioni DATA per tutti i caratteri, ma solo per quelli che avete cambiato. Dopo aver premuto CTRL-D il programma aggiunge i caratteri in coda a qualsiasi programma che avete nella memoria riservata al BASIC. Se non vi è un programma, esistono le sole istruzioni DATA. Potete caricare il programma, battere NEW per ri-inizializzare alcuni puntatori BASIC, caricare un programma su cui state lavorando, quindi attivare l'«Assortimento completo di caratteri da stampa», con una SYS 49152, per aggiungere le istruzioni DATA alla fine del programma. Le istruzioni DATA partono sempre dalla riga 60000, quindi potreste aver bisogno di cambiarne la numerazione. Se premete per due volte CTRL-D, un'altra serie di istruzioni DATA verrà aggiunta, anch'essa a partire dalla riga 60000 e seguenti. Fate riferimento alle note al termine dell'articolo per avere maggiori dettagli sull'uso delle istruzioni DATA nei vostri programmi.

Come si esce dal programma

Dopo aver creato le frasi DATA vi troverete ancora all'interno del

programma. Se volete uscirne, per vedere le istruzioni DATA o passare ad altro, premete CTRL-X. Lo schermo verrà ripristinato ai colori normali e vedrete apparire la scritta READY. Se avete creato delle frasi DATA, una istruzione LIST lo rivelerà immediatamente. Vi raccomando di battere il comando CLR per assicurarvi che il BASIC sia inizializzato opportunamente dopo l'uscita dal programma. Una cosa a cui prestare attenzione: non usate RUN/STOP-RESTORE per uscire dal programma. Il programma, infatti, sposta la memoria di schermo dalla sua area standard in 1024 e la combinazione di tasti RUN/STOP-RESTORE non ripristina i puntatori del sistema operativo alla memoria di schermo. Se li premete, non sarete in grado di vedere i tasti che state battendo. Per ovviare all'inconveniente battete alla cieca POKE 648,4 o SYS 49152 per rientrare nel programma, in modo da poterne uscire correttamente.

Come si rientra nel programma

Dopo esserne usciti potete rieseguire il programma con SYS 49152. Rivedrete il set di caratteri su cui stavate lavorando in precedenza, insieme al messaggio USE ROM SET (Y/N). Normalmente il programma copierà le forme contenute nella ROM carattere in RAM, dove potete cambiarle. Se premete N, tuttavia, il set su cui stavate lavorando in precedenza rimarrà intatto. Premete qualsiasi altro tasto, come RETURN, per ripristinare i caratteri allo standard contenuto nella ROM.

Un pianeta completamente diverso

Non abbiamo ancora finito. C'è un modo completamente diverso di operare con il nostro programma, il formato multicolore, nel quale qualsiasi carattere può contenere fino a quattro colori (compreso il colore di fondo) contemporaneamente. Il formato multicolore cambia il modo stesso in cui il calcolatore interpreta la forma dei caratteri. Invece di considerare i bit posti a 1 come pixel attivati e quelli a 0 come spazi vuoti, gli otto bit vengono organizzati in quattro coppie di bit. Ciascuna coppia può presentare quattro possibilità: 00, 01, 10 e 11. Anche ciascuna di queste è un numero decimale compreso tra 0 e 3. Ciascuna configurazione di bit di due bit rappresenta uno dei quattro colori. (La programmazione e l'utilizzo dei caratteri multicolori sono descritti nell'articolo che segue:

«Un uso evoluto di caratteri grafici»).

Il programma rende semplice il formato multicolore. Non dovete badare alle coppie di bit, non dovete più operare conversioni da binario in decimale. Limitatevi a premere il tasto funzione f5. Ed ecco, l'intero schermo cambia. I caratteri sono praticamente irriconoscibili ed il cursore che modifica i caratteri ha dimensioni doppie, dal momento che otto bit sono stati ridotti a quattro coppie di pixel, rendendo ciascun punto grande il doppio. Avete solo quattro punti per carattere, orizzontalmente, ma potete facilmente combinare più caratteri per formare figure più grandi.

Il formato multicolore ridefinisce il modo in cui funzionano il joystick ed il pulsante di sparo. Il pulsante di sparo traccia un rettangolo colorato nel colore con cui state attualmente lavorando. Il colore che esso traccia viene mostrato al centro del cursore che modifica i caratteri. Premete i tasti numerici 1, 2, 3 o 4 per scegliere i differenti colori con cui lavorare. Il numero dei tasti è maggiore di uno rispetto alla configurazione dei bit, quindi il colore 1 corrisponde alla configurazione di bit 00 ed il colore 4 alla 11. Quando attivate il programma con una SYS, i quattro colori sono nero (colore di fondo), bianco, azzurro e porpora. Questi quattro colori si distinguono facilmente anche su un televisore o su un monitor in bianco e nero.

Potete facilmente cambiare i colori. Dovete solo tenere premuto SHIFT e battere il tasto numerico opportuno per cambiare il colore associato a quel numero. Vedrete apparire il messaggio PRESS COLOR KEY. Ora premete uno dei tasti colore che vanno da CTRL-1 a CTRL-8 o da Commodore-1 a Commodore-8. Tenete premuto CTRL od il tasto con il marchio Commodore mentre lo fate.

Il colore indicato e qualsiasi cosa sia stata precedentemente tracciata in quel colore vengono immediatamente cambiati.

Tre dei colori (compreso il numero 1, il colore di fondo) possono essere uno qualsiasi dei 16 colori disponibili. Ma, a causa del funzionamento del formato multicolore, il colore 4, rappresentato dalla configurazione di bit 11, o 3 in valore decimale, può essere solo uno degli otto colori ottenibili con il tasto CTRL. Assegnargli uno dei colori ottenibili con il tasto recante il marchio Commodore provocherà semplicemente l'assunzione del colore relativo al tasto CTRL. Detto per inciso, è il colore della configurazione di bit 3 (il colore 4) che cambia, in base al colore del carattere come viene definito internamente alla memoria colore. Gli altri colori sono

programmati nei registri multicolore 1 e 2 (POKE 53282 e 53283), quindi tutti i caratteri condividono questi due colori. Quando volete cambiare un determinato colore senza influenzare il resto dei caratteri dovrete inserirlo come colore 4.

Alcuni dei comandi in formato multicolore non sono altrettanto utili di altri. Dovete premere f1 ed f2 due volte per far scorrere un carattere, dal momento che fanno scorrere un solo bit, il che provoca un cambiamento di colori. Potete usare CTRL-R, REVERSE, per porre in negativo tutti i colori (il colore 1 diventa colore 4, il colore 2 diventa colore 3 ed il colore 3 diventa colore 2). R: il comando di rotazione cambia tutti i colori ed è praticamente inutile, a meno che lo premiate due volte in modo da capovolgere semplicemente il carattere. M: il comando di riflessione invertirà i colori 2 e 3, dal momento che la configurazione di bit 01 (colore 2) diventa 10 (colore 3). Potete ancora copiare i caratteri utilizzando f7 e f8 (vedi più sopra).

Come si ritorna alla normalità

Potete riattivare immediatamente il formato standard dei caratteri, premendo f6 (SHIFT-f5). Se stavate disegnando in multicolore, potete vedere la configurazione di bit che rappresentava ciascun colore. In altre parole, i caratteri multicolore sembrano altrettanto strani in formato normale di quanto i caratteri in formato normale sembrino in multicolore.

Se avete cambiato colori in formato multicolore, alcuni dei colori in formato normale possono essere diversi. Potete cambiare questi colori come in formato multicolore. Premete SHIFT-1 per cambiare il colore dei pixel vuoti e SHIFT-3 per cambiare il colore delle otto file di caratteri. Usate SHIFT-2 per cambiare il colore dei pixel attivati.

Programmazione

Il paragrafo seguente vi mostra come potete utilizzare al meglio i caratteri. Comprende parecchi brevi programmi di utilità pratica in linguaggio macchina, che potete usare quando progettate giochi od altri programmi che utilizzano questi caratteri personalizzati. Mostra come il vostro programma possa leggere direttamente i file registrati, senza dover utilizzare istruzioni POKE da frasi DATA. Dovreste anche aver ben afferrato i concetti fondamentali della

programmazione dei caratteri (vedere il primo articolo di questo capitolo: «Costruitevi i vostri caratteri personalizzati»). Questo programma è pensato come un ausilio artistico alla vostra capacità creativa: è più conveniente lasciare che il calcolatore si occupi dei compiti più noiosi.

Note: come usare le istruzioni DATA

Le istruzioni DATA vengono create a partire dalla riga 60000, nel numero che si rende necessario. Il primo numero è il codice interno del carattere (il codice simbolico che utilizzate quando inserite un carattere sullo schermo tramite una POKE). Rappresenta uno scarto all'interno della tavola contenente le forme dei caratteri. Gli otto numeri che seguono rappresentano i valori numerici decimali per gli otto byte richiesti per definire ciascun carattere. Un programma esemplificativo per leggerli e mostrarli potrebbe essere:

```
10 POKE 52,48:POKE 56,48:CLR
50 READ A:IF A=-1 THEN 70
60 FOR I=0 TO 7:READ B:POKE
   12288+A*8+I,B:NEXT:GOTO 50
70 PRINT CHR$(147);"[10 GIU']":
   REM PREMERE DIECI VOLTE IL
   TASTO CRSR DI DESTRA
80 FOR I=0 TO 7:FOR J=0TO31:
   POKE 1028+J+I*40,I*32+J:POKE
    55300+J+I*40,1:NEXT:NEXT
90 POKE 53272,(PEEK(53272) AND
    240) OR 12:END
```

Aggiungete alle frasi DATA:
63999 DATA — 1

Se volete anche le forme normali contenute nella ROM, potete copiarle in RAM aggiungendo:

```
20 POKE 56334,PEEK(56334) AND 254
   :POKE 1,PEEK(1) AND 251
30 FOR I=0 TO 2047:POKE 12288+I,
   PEEK(53248+I):NEXT
40 POKE 1,PEEK(1) OR 4:POKE 56334
   ,PEEK(56334) OR 1
```

Capitolo uno

I comandi del programma (joystick nella Port 2)

Tasti cursore:	Si spostano sul carattere adiacente.
HOME (CLR/HOME):	Sposta il cursore nell'angolo superiore sinistro. Premetelo due volte per ritornare all'inizio del programma.
V:	Velocità del cursore. Date una risposta da 0 (lento) a 9 (veloce).
f1:	Scorrimento verso destra con ricomparsa.
f2(SHIFT-F1):	Scorrimento verso sinistra.
f3:	Scorrimento verso il basso.
f4:(SHIFT-f3):	Scorrimento verso l'alto.
R:	Rotazione di 90 gradi. Premete due volte per invertire i caratteri.
M:	Immagine riflessa.
SHIFT CLR/HOME:	Cancella il carattere corrente.
CTRL-R, CTRL-9:	Pone i pixel in negativo.
F:	Ripristina la forma dei caratteri ricavandola dalla ROM.
L:	Caricamento. Nastro o Disco, Nome del File.
S:	Registrazione. Nastro o Disco, Nome del File.
f7:	Memorizza un carattere (lo trattiene in un'area tampone).
f8:(SHIFT-f7):	Richiama il carattere (lo inserisce prendendolo dall'area tampone).

Un set completo di caratteri da stampa

```
49152 :076,200,196,000,001,003,220
49158 :004,000,173,048,002,072,049
49164 :173,045,002,141,048,002,167
49170 :141,079,002,032,043,193,252
49176 :104,141,048,002,169,100,076
49182 :133,252,169,000,133,251,200
49188 :133,167,169,216,133,168,254
49194 :169,008,141,040,002,169,059
49200 :002,141,042,002,169,005,153
```

```
49206  :141,041,002,174,003,192,095
49212  :173,079,002,205,048,002,057
49218  :208,002,162,006,142,080,154
49224  :002,160,000,177,253,170,066
49230  :173,063,002,240,003,076,123
49236  :229,192,169,207,145,251,253
49242  :138,010,170,176,008,173,253
49248  :080,002,145,167,076,108,162
49254  :192,173,004,192,145,167,207
49260  :200,192,008,208,221,024,193
49266  :165,251,105,008,133,251,003
49272  :133,167,165,252,105,000,174
49278  :133,252,105,116,133,168,009
49284  :024,165,253,105,008,133,052
49290  :253,165,254,105,000,133,024
49296  :254,056,238,079,002,206,211
49302  :041,002,173,041,002,208,105
49308  :156,056,173,079,002,233,087
49314  :005,141,079,002,056,165,098
49320  :253,233,039,133,253,165,220
49326  :254,233,000,133,254,206,230
49332  :040,002,173,040,002,240,165
49338  :003,076,052,192,206,042,245
49344  :002,173,042,002,240,030,169
49350  :169,008,141,040,002,024,070
49356  :173,079,002,105,032,141,224
49362  :079,002,024,165,253,105,070
49368  :248,133,253,165,254,105,094
49374  :000,133,254,076,052,192,161
49380  :096,134,097,169,000,141,097
49386  :043,002,006,097,046,043,215
49392  :002,006,097,046,043,002,180
49398  :174,043,002,169,207,145,218
49404  :251,200,169,247,145,251,235
49410  :136,189,003,192,145,167,066
49416  :200,145,167,200,192,008,152
49422  :208,215,076,113,192,169,219
49428  :000,141,026,208,165,001,049
49434  :041,251,133,001,096,165,201
```

```
49440  :001,009,004,133,001,169,093
49446  :001,141,026,208,096,169,167
49452  :000,133,254,173,048,002,142
49458  :133,253,006,253,038,254,219
49464  :006,253,038,254,006,253,098
49470  :038,254,024,169,112,101,248
49476  :254,133,254,096,032,043,112
49482  :193,160,000,177,253,073,162
49488  :255,145,253,200,192,008,109
49494  :208,245,032,008,192,096,099
49500  :169,102,133,252,169,218,111
49506  :133,168,169,132,133,251,060
49512  :133,167,162,008,169,000,231
49518  :133,097,160,000,165,097,250
49524  :145,251,230,097,173,058,046
49530  :002,145,167,200,192,032,092
49536  :208,240,024,165,251,105,097
49542  :040,133,251,133,167,165,255
49548  :252,105,000,133,252,105,219
49554  :116,133,168,202,208,216,165
49560  :096,032,043,203,173,044,231
49566  :002,141,024,208,169,200,134
49572  :013,063,002,141,022,208,101
49578  :169,000,141,032,208,141,093
49584  :033,208,032,092,193,173,139
49590  :058,002,141,134,002,165,172
49596  :209,133,243,024,165,210,148
49602  :105,116,133,244,164,211,143
49608  :177,209,073,128,145,209,117
49614  :177,243,072,173,134,002,239
49620  :145,243,032,228,255,240,075
49626  :251,170,164,211,177,209,120
49632  :073,128,145,209,104,145,004
49638  :243,138,032,210,255,032,116
49644  :225,255,208,203,032,075,210
49650  :203,169,000,141,134,002,123
49656  :169,012,141,032,208,076,118
49662  :094,196,032,019,193,169,189
49668  :112,133,252,173,082,002,246
```

```
49674  :133,254,162,008,169,000,224
49680  :133,253,133,251,168,177,107
49686  :253,145,251,200,208,249,048
49692  :230,254,230,252,202,208,124
49698  :242,165,252,201,128,240,238
49704  :007,169,208,133,254,076,119
49710  :012,194,032,031,193,096,092
49716  :169,112,133,252,169,116,235
49722  :133,254,169,000,133,253,232
49728  :133,251,168,162,004,177,191
49734  :251,073,255,145,253,200,223
49740  :208,247,230,254,230,252,217
49746  :202,208,240,096,032,043,135
49752  :193,160,000,177,253,010,113
49758  :008,074,040,042,145,253,144
49764  :200,192,008,208,242,076,002
49770  :008,192,032,043,193,160,222
49776  :000,177,253,074,008,010,122
49782  :040,106,145,253,200,192,030
49788  :008,208,242,076,008,192,090
49794  :032,043,193,160,000,177,223
49800  :253,133,097,200,177,253,225
49806  :136,145,253,200,200,192,244
49812  :008,208,245,165,097,136,239
49818  :145,253,076,008,192,032,092
49824  :043,193,160,007,177,253,225
49830  :133,097,136,177,253,200,138
49836  :145,253,136,016,247,200,145
49842  :165,097,145,253,076,008,154
49848  :192,032,043,193,160,000,036
49854  :169,000,133,097,162,008,247
49860  :177,253,010,102,097,202,013
49866  :208,250,165,097,145,253,040
49872  :200,192,008,208,233,076,101
49878  :008,192,032,043,193,160,074
49884  :008,169,000,153,048,002,088
49890  :136,208,250,169,007,133,105
49896  :097,152,170,169,000,133,185
49902  :007,177,253,074,145,253,123
```

```
49908  :038,007,202,016,251,166,156
49914  :097,165,007,029,049,002,087
49920  :157,049,002,198,097,165,156
49926  :097,016,224,200,192,008,231
49932  :208,215,136,185,049,002,039
49938  :145,253,136,016,248,076,124
49944  :008,192,032,043,193,160,140
49950  :000,152,145,253,200,192,204
49956  :008,208,249,076,008,192,009
49962  :120,169,127,141,013,220,064
49968  :169,001,141,026,208,169,250
49974  :177,141,018,208,169,027,026
49980  :141,017,208,169,075,141,043
49986  :020,003,169,195,141,021,103
49992  :003,088,096,173,018,208,146
49998  :201,177,208,039,169,242,090
50004  :141,018,208,173,044,002,158
50010  :141,024,208,173,022,208,098
50016  :041,239,013,063,002,141,083
50022  :022,208,173,057,002,141,193
50028  :033,208,169,001,141,025,173
50034  :208,104,168,104,170,104,204
50040  :064,169,177,141,018,208,129
50046  :169,158,141,024,208,173,231
50052  :032,208,141,033,208,169,155
50058  :200,141,022,208,238,037,216
50064  :208,169,001,141,025,208,128
50070  :076,049,234,085,064,000,146
50076  :064,064,000,076,064,000,168
50082  :076,064,000,076,064,000,186
50088  :076,064,000,064,064,000,180
50094  :085,064,000,000,000,085,152
50100  :080,000,064,016,000,064,148
50106  :016,000,064,016,000,064,090
50112  :016,000,064,016,000,064,096
50118  :016,000,064,016,000,064,102
50124  :016,000,085,080,000,000,129
50130  :000,000,255,255,255,000,207
50136  :001,001,001,000,255,001,219
```

```
50142  :000,000,255,001,000,000,222
50148  :255,001,018,085,076,084,235
50154  :082,065,070,079,078,084,180
50160  :032,043,146,095,069,082,195
50166  :082,079,082,032,079,078,166
50172  :032,083,065,086,069,047,122
50178  :076,079,065,068,095,018,147
50184  :084,146,065,080,069,032,228
50190  :079,082,032,018,068,146,183
50196  :073,083,075,063,095,070,223
50202  :073,076,069,078,065,077,208
50208  :069,058,095,069,078,084,229
50214  :069,082,032,067,079,076,187
50220  :079,082,032,075,069,089,214
50226  :095,085,083,069,032,082,240
50232  :079,077,032,083,069,084,224
50238  :063,032,040,089,047,078,155
50244  :041,095,018,085,146,080,021
50250  :080,069,082,067,065,083,008
50256  :069,032,079,082,032,018,136
50262  :076,146,079,087,069,082,113
50268  :063,095,169,230,160,195,236
50274  :133,251,132,252,160,040,042
50280  :169,032,153,191,103,136,120
50286  :208,250,177,251,200,201,117
50292  :095,208,249,136,132,097,009
50298  :152,074,073,255,056,105,069
50304  :020,168,162,024,024,032,046
50310  :240,255,160,000,177,251,193
50316  :032,210,255,200,196,097,106
50322  :144,246,096,133,251,132,124
50328  :252,160,040,169,032,153,190
50334  :191,103,136,208,250,162,184
50340  :024,160,000,024,032,240,132
50346  :255,160,000,177,251,201,190
50352  :095,240,006,032,210,255,246
50358  :200,208,244,096,174,076,156
50364  :002,240,008,160,000,200,030
50370  :208,253,202,208,250,096,131
```

```
50376  :173,002,221,009,003,141,237
50382  :002,221,173,000,221,041,096
50388  :252,009,002,141,000,221,069
50394  :169,100,141,136,002,169,167
50400  :147,032,210,255,169,000,013
50406  :141,134,002,169,008,032,204
50412  :210,255,160,000,152,153,142
50418  :128,099,200,016,250,168,079
50424  :185,153,195,153,128,099,137
50430  :200,192,023,208,245,160,002
50436  :000,185,176,195,153,192,137
50442  :099,200,192,032,208,245,218
50448  :169,156,141,044,002,169,185
50454  :012,141,032,208,169,128,200
50460  :141,138,002,032,042,195,066
50466  :169,048,141,076,002,169,127
50472  :011,141,057,002,169,007,171
50478  :169,000,141,048,002,141,035
50484  :045,002,141,063,002,173,222
50490  :006,192,009,008,141,058,216
50496  :002,173,004,192,141,034,098
50502  :208,173,005,192,141,035,056
50508  :208,032,008,192,032,092,128
50514  :193,169,203,205,007,192,027
50520  :240,014,141,007,192,162,076
50526  :208,142,082,002,032,000,048
50532  :194,076,120,197,169,051,139
50538  :160,196,032,098,196,032,052
50544  :228,255,240,251,201,078,085
50550  :240,029,169,070,160,196,214
50556  :032,098,196,032,228,255,197
50562  :240,251,162,208,201,076,244
50568  :208,002,162,216,142,082,180
50574  :002,032,000,194,032,008,154
50580  :192,032,094,196,169,142,205
50586  :141,248,103,169,143,141,075
50592  :249,103,169,003,141,021,078
50598  :208,169,024,141,000,208,148
50604  :169,000,141,016,208,169,107
```

```
50610  :051,141,001,208,169,176,156
50616  :141,003,208,169,053,141,131
50622  :002,208,169,000,141,029,227
50628  :208,141,023,208,141,038,187
50634  :208,169,003,141,028,208,191
50640  :169,000,141,059,002,141,208
50646  :060,002,173,000,220,072,229
50652  :041,015,073,015,141,061,054
50658  :002,104,041,016,141,062,080
50664  :002,032,228,255,240,006,227
50670  :032,109,199,076,216,197,043
50676  :032,186,196,173,062,002,127
50682  :208,003,032,000,199,173,097
50688  :062,002,073,016,141,075,113
50694  :002,173,061,002,240,204,176
50700  :174,061,002,189,208,195,073
50706  :172,063,002,240,001,010,250
50712  :024,109,059,002,141,059,162
50718  :002,024,173,060,002,125,160
50724  :219,195,141,060,002,174,059
50730  :059,002,016,025,162,000,050
50736  :142,059,002,173,045,002,215
50742  :240,015,206,045,002,162,212
50748  :007,173,063,002,240,002,035
50754  :162,006,142,059,002,174,099
50760  :059,002,224,040,144,022,051
50766  :162,039,142,059,002,173,143
50772  :045,002,201,219,176,010,225
50778  :105,001,141,045,002,162,034
50784  :032,142,059,002,172,060,051
50790  :002,016,022,160,000,140,186
50796  :060,002,173,045,002,201,079
50802  :032,144,010,233,032,141,194
50808  :045,002,160,007,140,060,022
50814  :002,172,060,002,192,016,058
50820  :144,022,160,015,140,060,161
50826  :002,173,045,002,201,192,241
50832  :176,010,105,032,141,045,141
50838  :002,160,008,140,060,002,010
```

```
50844  :173,059,002,172,060,002,112
50850  :074,074,074,192,008,144,216
50856  :002,105,031,109,045,002,206
50862  :141,048,002,041,224,074,192
50868  :074,105,176,141,003,208,119
50874  :173,048,002,041,031,010,235
50880  :010,010,105,053,141,002,001
50886  :208,169,000,105,000,133,045
50892  :097,173,060,002,010,010,044
50898  :010,105,051,141,001,208,214
50904  :173,059,002,010,010,010,224
50910  :038,097,105,024,141,000,115
50916  :208,165,097,105,000,141,176
50922  :016,208,173,048,002,205,118
50928  :081,002,240,009,032,008,100
50934  :192,173,048,002,141,081,115
50940  :002,076,216,197,032,043,050
50946  :193,173,060,002,041,007,222
50952  :168,173,059,002,041,007,202
50958  :073,007,170,232,134,097,215
50964  :056,169,000,042,202,208,185
50970  :252,174,063,002,208,048,005
50976  :133,097,173,075,002,208,208
50982  :022,169,000,141,064,002,180
50988  :141,038,208,177,253,037,130
50994  :097,208,008,169,001,141,162
51000  :064,002,141,038,208,165,162
51006  :097,073,255,049,253,174,195
51012  :064,002,240,002,005,097,222
51018  :145,253,032,008,192,096,032
51024  :133,098,074,005,098,073,049
51030  :255,049,253,166,097,202,084
51036  :133,097,173,066,002,074,125
51042  :042,202,208,252,005,097,136
51048  :145,253,076,008,192,141,151
51054  :065,002,174,137,199,221,140
51060  :137,199,240,004,202,208,082
51066  :248,096,202,138,010,170,218
51072  :189,173,199,072,189,172,098
```

```
51078  :199,072,096,034,133,137,037
51084  :134,138,077,082,147,018,224
51090  :145,017,157,029,070,135,187
51096  :139,049,050,051,052,019,000
51102  :136,140,033,034,035,036,060
51108  :086,083,076,024,004,006,187
51114  :131,084,107,194,085,194,197
51120  :158,194,129,194,184,194,205
51126  :215,194,025,195,071,193,051
51132  :248,199,014,200,036,200,061
51138  :058,200,082,200,117,200,027
51144  :160,200,172,200,172,200,024
51150  :172,200,172,200,189,200,059
51156  :214,200,236,200,014,201,253
51162  :014,201,014,201,014,201,095
51168  :085,201,136,202,020,203,047
51174  :036,203,160,203,051,194,053
51180  :239,199,152,193,162,255,156
51186  :154,032,129,255,076,200,064
51192  :196,173,060,002,041,007,215
51198  :133,097,056,173,060,002,007
51204  :233,008,056,229,097,141,000
51210  :060,002,076,078,200,173,087
51216  :060,002,041,007,133,097,100
51222  :024,173,060,002,105,008,138
51228  :056,229,097,141,060,002,101
51234  :076,078,200,173,059,002,110
51240  :041,007,133,097,056,173,035
51246  :059,002,233,008,056,229,121
51252  :097,141,059,002,076,078,249
51258  :200,173,059,002,041,007,028
51264  :133,097,024,173,059,002,040
51270  :105,008,056,229,097,141,194
51276  :059,002,104,104,076,041,206
51282  :198,032,043,193,032,019,087
51288  :193,160,007,024,165,254,123
51294  :105,096,141,106,200,165,139
51300  :253,141,105,200,185,000,216
51306  :208,145,253,136,016,248,088
```

```
51312 :032,031,193,076,008,192,132
51318 :169,016,141,063,002,169,166
51324 :001,141,029,208,032,008,031
51330 :192,173,058,002,009,008,060
51336 :141,058,002,032,092,193,142
51342 :169,050,141,065,002,032,089
51348 :173,200,173,059,002,041,028
51354 :254,141,059,002,076,078,252
51360 :200,169,000,141,063,002,223
51366 :141,029,208,032,008,192,008
51372 :096,056,173,065,002,233,029
51378 :049,141,066,002,170,189,027
51384 :003,192,141,038,208,096,094
51390 :173,059,002,013,060,002,243
51396 :208,003,141,045,002,169,252
51402 :000,141,059,002,141,060,093
51408 :002,032,008,192,076,078,084
51414 :200,032,072,193,032,072,047
51420 :193,032,043,193,160,000,073
51426 :177,253,153,067,002,200,054
51432 :192,008,208,246,096,032,246
51438 :043,193,160,000,185,067,118
51444 :002,145,253,200,192,008,020
51450 :208,246,076,008,192,144,100
51456 :005,028,159,156,030,031,153
51462 :158,129,149,150,151,152,127
51468 :153,154,155,169,035,160,070
51474 :196,032,098,196,032,228,032
51480 :255,240,251,162,000,221,129
51486 :255,200,240,008,232,224,165
51492 :016,208,246,076,094,196,104
51498 :056.173,065,002,233,033,092
51504 :168,138,153,003,192,192,126
51510 :003,240,010,192,000,240,227
51516 :019,153,033,208,076,080,117
51522 :201,174,063,002,240,002,236
51528 :009,008,141,058,002,032,066
51534 :092,193,032,008,192,076,159
51540 :094,196,169,122,160,201,002
```

```
51546  :032,098,196,032,228,255,163
51552  :056,233,048,048,248,201,162
51558  :010,176,244,133,097,056,050
51564  :169,009,229,097,010,010,120
51570  :010,010,141,076,002,076,173
51576  :094,196,067,085,082,083,215
51582  :079,082,032,086,069,076,038
51588  :079,067,073,084,089,032,044
51594  :040,048,045,057,041,063,176
51600  :095,160,000,140,078,002,107
51606  :169,164,032,210,255,169,125
51612  :157,032,210,255,032,228,046
51618  :255,240,251,172,078,002,136
51624  :133,097,169,032,032,210,073
51630  :255,169,157,032,210,255,228
51636  :165,097,201,013,240,039,167
51642  :201,020,208,013,192,000,052
51648  :240,209,136,169,157,032,111
51654  :210,255,076,147,201,041,104
51660  :127,201,032,144,194,192,070
51666  :020,240,190,165,097,153,051
51672  :000,002,032,210,255,200,147
51678  :076,147,201,169,095,153,039
51684  :000,002,152,096,032,231,229
51690  :255,169,007,160,196,032,029
51696  :098,196,032,228,255,240,009
51702  :251,162,001,201,084,240,161
51708  :011,162,008,201,068,240,174
51714  :005,104,104,076,094,196,069
51720  :141,077,002,160,001,169,046
51726  :001,032,186,255,169,025,170
51732  :160,196,032,149,196,032,017
51738  :145,201,208,007,173,077,069
51744  :002,201,084,208,237,173,169
51750  :077,002,201,068,208,066,148
51756  :169,064,141,020,002,169,097
51762  :048,141,021,002,169,058,233
51768  :141,022,002,160,000,185,054
51774  :000,002,153,023,002,200,186
```

```
51780  :204,078,002,208,244,169,205
51786  :044,153,023,002,169,080,033
51792  :153,024,002,173,065,002,243
51798  :201,083,208,012,169,044,035
51804  :153,025,002,169,087,153,169
51810  :026,002,200,200,200,200,158
51816  :200,200,200,076,124,202,082
51822  :160,000,185,000,002,153,098
51828  :020,002,200,204,078,002,110
51834  :208,244,152,162,020,160,044
51840  :002,032,189,255,169,160,167
51846  :133,178,096,032,232,201,238
51852  :032,043,203,169,000,133,208
51858  :253,133,251,169,112,133,173
51864  :252,162,255,160,119,169,245
51870  :251,032,216,255,176,011,075
51876  :032,183,255,208,006,032,112
51882  :075,203,076,094,196,032,078
51888  :075,203,032,231,255,173,121
51894  :077,002,201,068,240,015,017
51900  :169,244,160,195,032,098,062
51906  :196,032,228,255,240,251,116
51912  :076,094,196,169,000,032,255
51918  :189,255,169,015,162,008,236
51924  :160,015,032,186,255,032,124
51930  :192,255,162,015,032,198,048
51936  :255,160,000,032,207,255,109
51942  :201,013,240,007,153,000,076
51948  :002,200,076,227,202,169,088
51954  :095,153,000,002,032,204,216
51960  :255,169,000,160,002,032,098
51966  :098,196,162,015,032,201,190
51972  :255,169,073,032,210,255,230
51978  :169,013,032,210,255,032,209
51984  :231,255,076,195,202,032,239
51990  :232,201,032,043,203,169,134
51996  :000,032,213,255,176,141,077
52002  :076,075,203,169,004,141,190
52008  :136,002,000,120,169,000,211
```

```
52014  :141,026,208,169,255,141,218
52020  :013,220,169,049,141,020,152
52026  :003,169,234,141,021,003,117
52032  :169,000,141,021,208,169,004
52038  :147,088,076,210,255,032,110
52044  :042,195,169,003,141,021,135
52050  :208,032,008,192,032,092,134
52056  :193,076,094,196,248,169,040
52062  :000,141,000,001,141,001,122
52068  :001,224,000,240,021,202,020
52074  :024,173,000,001,105,001,154
52080  :141,000,001,173,001,001,173
52086  :105,000,141,001,001,076,186
52092  :101,203,216,173,001,001,051
52098  :009,048,141,002,001,173,248
52104  :000,001,041,240,074,074,054
52110  :074,074,009,048,141,001,233
52116  :001,173,000,001,041,015,123
52122  :009,048,141,000,001,096,193
52128  :096,056,165,045,233,002,245
52134  :133,045,165,046,233,000,020
52140  :133,046,169,024,133,057,222
52146  :169,246,133,058,169,000,185
52152  :141,079,002,133,251,133,155
52158  :253,169,112,133,254,169,000
52164  :208,133,252,032,019,193,009
52170  :160,000,177,251,209,253,228
52176  :208,058,200,192,008,208,058
52182  :245,238,079,002,024,165,199
52188  :253,105,008,133,253,133,081
52194  :251,165,254,105,000,133,110
52200  :254,105,096,133,252,201,249
52206  :216,208,217,169,000,168,192
52212  :145,045,200,145,045,024,080
52218  :165,045,105,002,133,045,233
52224  :165,046,105,000,133,046,239
52230  :032,031,193,076,051,165,042
52236  :160,000,024,165,045,105,255
52242  :041,145,045,200,165,046,148
```

```
52248 :105,000,145,045,200,165,172
52254 :057,145,045,200,165,058,188
52260 :145,045,200,169,131,145,103
52266 :045,174,079,002,032,092,210
52272 :203,200,173,002,001,145,004
52278 :045,200,173,001,001,145,107
52284 :045,200,173,000,001,145,112
52290 :045,200,132,097,160,000,188
52296 :132,098,177,253,170,032,166
52302 :092,203,164,097,169,044,079
52308 :145,045,200,173,002,001,138
52314 :145,045,173,001,001,200,143
52320 :145,045,173,000,001,200,148
52326 :145,045,200,132,097,164,117
52332 :098,200,192,008,208,214,004
52338 :164,097,169,000,145,045,222
52344 :160,000,177,045,072,200,006
52350 :177,045,133,046,104,133,252
52356 :045,230,057,208,002,230,136
52362 :058,076,215,203,022,237,181
```

Capitolo uno

Un uso evoluto dei caratteri grafici

Charles Brannon

Le numerose capacità grafiche offerte dal Commodore 64 sono sufficienti a lasciare stupefatto qualsiasi programmatore. Sprite, 16 colori, alta risoluzione: c'è di tutto. Ma parecchie persone hanno tralasciato una delle tecniche grafiche più potenti: la personalizzazione dei caratteri.

I caratteri personalizzati sono molto pratici nei giochi. Potete ridefinire qualsiasi lettera dell'alfabeto e spostare il carattere lungo lo schermo con semplici istruzioni POKE o PRINT. Potete persino programmare l'alfabeto di una lingua straniera od una serie particolare di simboli tecnici o matematici.

Normalmente si usano i caratteri personalizzati nel formato di testo normale, ma esistono numerose variazioni. La più interessante è rappresentata dai caratteri multicolori. Normalmente un carattere è composto solo da un colore e dal colore di fondo. Il carattere può essere in uno qualsiasi dei 16 colori di testo. In formato multicolore un singolo carattere può essere definito con tre colori, più il colore di fondo. Tutto sta nel modo in cui si definisce il carattere.

Nel formato di testo normale una configurazione binaria stabilisce la forma del carattere. Ciascuna delle otto righe di un carattere è definita da un byte binario di otto bit: un bit per colonna. La rappresentazione binaria considera gli 1 come pixel, o punti, attivati e rileva gli 0 come riquadri vuoti. In questo modo si costruisce l'intero carattere.

Il formato multicolore «raggruppa» i bit per consentire più di un colore per pixel. Invece di 1, che rappresentano pixel illuminati, e 0, che rappresentano spazi vuoti, i bit sono accoppiati in modo da formare quattro configurazioni binarie: 00, 01, 10 e 11. Ciascuna

configurazione binaria rappresenta un numero decimale da 0 a 3. Potete quindi avere quattro colori (quattro possibili coppie di bit), ma la risoluzione orizzontale viene dimezzata, dal momento che il chip VIC-II continua ad utilizzare un solo byte per riga.

Questa limitazione può essere facilmente superata combinando più caratteri, in modo da formare una figura più grande. Ciascun carattere ha una risoluzione di quattro punti colorati orizzontalmente e di otto verticalmente. Potete raggruppare i caratteri in una disposizione 2x2, in modo da ottenere una matrice 8x16, o in una configurazione 3x2 per ottenere una griglia 12x16. Il programma per la costruzione di caratteri personalizzati dell'articolo precedente vi permette di disegnare facilmente figure più grandi di un carattere. Potete anche definire dei blocchi da costruzione, figure utilizzabili per costruire aree più vaste, come mattoni, scale o catene montuose. Potete usare caratteri multicolori per disegnare attraenti paesaggi per giochi che utilizzino gli sprite, senza dover utilizzare la pagina grafica ad alta risoluzione.

Qualche piccola informazione di carattere tecnico

Ecco come vengono predisposti i quattro colori. Ciascuna configurazione di bit rappresenta un codice che segnala la provenienza del colore. La configurazione di bit 00 è uno spazio vuoto, il quale ha l'aspetto del colore di fondo che viene mostrato in trasparenza. Il colore della configurazione 01 proviene dalla locazione 53282. Può contenere un qualsiasi valore da 0 a 15: tutti i punti tracciati nella configurazione di bit 01 avranno il colore corrispondente. La configurazione 10 (decimale 2) ricava nello stesso modo il proprio colore dal registro multicolore 53283. La configurazione 11 (decimale 3) ha un'origine diversa. Il suo colore proviene dal colore del carattere contenuto nella memoria colore. Ma potete usare solo i colori contraddistinti da un codice simbolico compreso tra 0 e 7.

Come si attiva il formato multicolore

Ecco perché: Il Commodore 64 vi consente di avere contemporaneamente sullo schermo sia caratteri normali che multicolori. Nel formato normale i colori contraddistinti da valori di codice simbolico compresi tra 8 e 15, ed a cui si accede tramite il tasto con il marchio Commodore, sono dei colori aggiunti, otto colori di testo

in più rispetto al VIC-20. In formato multicolore, invece, qualsiasi colore con un codice superiore a 7 indica una colorazione a più colori. Il colore della configurazione di bit 11 è rappresentato dai colori da 8 a 15 meno 8.

Un carattere in formato multicolore stampato in colore Commodore-1 non utilizzerà l'arancio, ma il nero (CTRL-1). In altre parole, i tre bit meno significativi del numero vengono utilizzati per predisporre il colore della configurazione multicolore di bit 11. Il bit 4 (decimale 8) segnala il formato multicolore e ciò impedisce l'uso dei colori Commodore.

Come scegliere una opzione

Il formato multicolore viene attivato predisponendo il bit 4 del registro 53270 del VIC-II. In BASIC dovreste battere:

```
10 POKE 53270,PEEK(53270) OR 16
```

Potete disattivare il formato multicolore con:

```
20 POKE 53270,PEEK(53270) AND 239
```

Dal momento che tutti i caratteri multicolori hanno gli stessi colori per le configurazioni di bit 01 e 10 (dai registri 53282 e 53283), generalmente desidererete usare la configurazione di bit 11 nel definire un carattere per i colori che volete cambiare. Ad esempio, qualsiasi nave spaziale da voi definita condividerà i due registri multicolore con le altre, ma potete avere una astronave blu ed una rossa, se programmate una nave spaziale con il colore della fusoliera nella configurazione di bit 11. Quindi, quando visualizzate l'astronave usate caratteri di differenti colori in memoria colore.

Potete, ovviamente, usare gli sprite contemporaneamente ai caratteri multicolori. Esiste un registro delle collisioni, che stabilisce se uno sprite ed un carattere di fondo si sono toccati. Il registro delle collisioni in 53279 ($D01F) contiene un valore numerico dal bit 0 al bit 7, che identifica quale sprite abbia colpito il carattere. Se lo sprite 3 (la numerazione parte da 0) ha colpito il carattere, il registro conterrà 2 alla terza potenza, cioè otto.

In formato multicolore le collisioni vengono generate nello stes-

so modo, ma l'hardware del calcolatore non rileva una collisione tra uno sprite ed una configurazione di bit 01. Potete rendere trasparenti (a prova di collisione) parti del paesaggio, programmando le forme con cui non è possibile entrare in collisione nella configurazione di bit 01. Potete quindi discriminare tra caratteri di due differenti colori, in base al fatto di avere o no ottenuto una collisione in 53279 ($D01F).

Il formato colore di fondo esteso

Questo formato, una eredità del VIC-20, vi fornisce quattro colori di fondo per carattere, in aggiunta ad uno qualsiasi dei 16 colori di primo piano. Potete utilizzarlo per evidenziare aree dello schermo in colori di fondo differenti, senza dover ricorrere a delle interruzioni di scansione video. Tuttavia potete usare solo i primi 64 caratteri del set *interno.*

Si attiva il formato colore di fondo esteso con:

```
POKE 53265,PEEK(53265) OR 64
```

Usate la riga seguente per disinserire il formato colore di fondo esteso:

```
POKE 53265,PEEK(53265) AND 191
```

Provate a battere qualche tasto, comprese le lettere maiuscole, i segni di interpunzione ed i caratteri grafici. Come potete vedere, ottenete differenti colori di fondo con ciascun carattere, ma avrete anche probabilmente notato che non avete accesso ai caratteri grafici. Ottenete solo il set di 64 caratteri alfanumerici standard, i primi 64 caratteri del set di caratteri interno.

Su 256 possibili codici simbolici di carattere (ottenibili con otto bit), solo cinque bit (vale a dire un valore numerico decimale compreso fra 0 e 63) vengono usati per indicare *quale* carattere visualizzare. I due bit più significativi (di valore, rispettivamente, 64 e 128) indicano di quale *colore* deve essere il fondo. Il colore del carattere stesso (colore di primo piano) viene definito dalla memoria colore, come al solito. Usate la tavola seguente (adattata da quella che compare sulla *Guida di riferimento del Programmatore*) per lavorare in formato colore di fondo esteso. Ricordatevi che i numeri si basano sui codici POKE dello schermo, non sul codice ASCII.

schermo/ Codice interno	Bit 7	Bit 6	Registro colore
Da: 0-63	0	0	53281
64-127	0	1	53282
128-191	1	0	53283
192-255	1	1	53284

Limitatevi a inserire, con una istruzione POKE, un numero di codice compreso tra 0 e 63 nella memoria di schermo ed il colore verrà ricavato dal registro 53281 (come avviene normalmente). Aggiungete 64 al numero di base ed il carattere ricaverà il suo colore di fondo dalla locazione 53282. Potete aggiungere sia 128 che 192 per ottenere gli altri due colori. Il programma seguente visualizza la lettera A in quattro colori di fondo:

```
10 POKE 53265,PEEK(53265) OR 64:
   BM(0)=0:BM(1)=64:BM(2)=128:BM
   (3)=192
20 POKE 53280,0:POKE 53281,0:POK
   E 53282,2:POKE 53283,8:POKE 5
   3284,7
30 PRINT"[CLR][WHT]e[RVS]St[OFF]
   Es[RVS]O"
40 FOR I=0 TO 15:OF=I*40:FOR J=0
    TO 3:POKE 1040+OF+J,1+BM(J):
   POKE 55312+OF+J,I:NEXT:NEXT
50 FOR W=1 TO 2000:NEXT:POKE 532
   65,PEEK(53265) AND 191:PRINT"
   [HOME]FORMATO NORMALE"
```

Il formato colore di fondo esteso utilizza alcuni degli stessi registri del formato multicolore. La Commodore suggerisce di non usare i due formati contemporaneamente. Non date loro retta: provate per credere!

Il resto dell'articolo è dedicato a parecchi brevi programmi in linguaggio macchina, che risultano particolarmente utili quando si programmano i caratteri. Potete usarli tutti contemporaneamente o separatamente. Ciascuno è costituito da una serie di istruzioni

DATA che potete aggiungere ai vostri programmi.

Richiamate le righe opportune con una istruzione GOSUB, in modo da leggere (con una istruzione READ) il codice del linguaggio macchina contenuto nelle istruzioni DATA ed inserirlo in memoria tramite delle istruzioni POKE. Usate quindi un comando SYS per richiamare ciascun programma.

Copia-ROM

Stabilite dove metterete il set di caratteri ed usate l'istruzione POKE adatta alla locazione 53272 per scegliere il banco di memoria da 2Kbyte che va utilizzato. Battete una istruzione SYS 49152 per copiare il set in maiuscolo della ROM nel banco della RAM che avete scelto. Cambiate il valore numerico 208 (in riga 49176) in 216, se preferite copiare il set in maiuscolo (se lo fate, naturalmente la somma di controllo in riga 49000 non coinciderà).

Programma 1. Copia-ROM

```
49000 FORI=49152TO49235:READA:CK=CK+A:POKE
      I,A:NEXT:IFCK=10131THENRETURN
49010 PRINT"{RVS}ERRORE NELLE FRASI DATA:
      CONTROLLALE":STOP
49152 DATA120,173,014,220,041,254
49158 DATA141,014,220,165,001,041
49164 DATA251,133,001,173,024,208
49170 DATA041,014,010,010,133,167
49176 DATA169,208,133,252,173,000
49182 DATA221,041,003,073,003,010
49188 DATA010,010,010,010,010,005
49194 DATA167,133,254,169,000,133
49200 DATA251,133,253,168,162,008
49206 DATA177,251,145,253,200,208
49212 DATA249,230,252,230,254,202
49218 DATA208,242,165,001,009,004
49224 DATA133,001,173,014,220,009
49230 DATA001,141,014,220,088,096
```

Come caricare il set di caratteri da nastro o da disco

Lo si può fare senza nessun altro programma in linguaggio macchina. Potete richiamare direttamente il sottoprogramma di caricamento della Kernal. Cambiate solo l'istruzione OPEN seguente, inserendo il nome sotto cui avete registrato il vostro set di caratteri

al posto di 'NOME DEL FILE'. Cambiate il valore numerico 8 in 1, se usate l'unità nastri. E cambiate CHSET, in modo che indirizzi il punto in cui volete che sia caricato il set di caratteri.

Progamma 2. Come caricare il set di caratteri

```
5000 REM CARICATORE DEL SET DI CARATTERI
5010 OPEN 1,8,8,"NOME DEL FILE"+",P,R"
5020 CHSET=14336:REM DOVE CARICARE
5030 POKE780,1:POKE781,1:POKE782,0:SYS 654
     66
5040 POKE780,0:POKE781,0:POKE782,CHSET/256
     :SYS 65493
5050 IFPEEK(783)AND1THENPRINT"ERRORE DI LE
     TTURA":STOP
5060 CLOSE1:RETURN
```

L'interruzione di cursore schermo (Raster interrupt)

Potete utilizzare questo programma per suddividere lo schermo in due aree, ciascuna con differenti set di caratteri e colori di fondo. Con esso potete riempire un set di caratteri con caratteri grafici ed utilizzare l'interruzione di cursore schermo per permettere alla riga in cui viene mostrato il punteggio di utilizzare i caratteri normali della ROM. Potete suddividere lo schermo in aree in maiuscolo ed in minuscolo. Potete persino avere due set di caratteri differenti nelle due aree.

Usate il comando SYS 49263 per inizializzare, quindi inserite nelle locazioni 831 e 832, con una istruzione POKE, il numero che avreste utilizzato per la locazione 53272 per ciascuno dei due set di caratteri (usate 21 per il set di caratteri standard, 28 per 12288). Inserite, sempre tramite una POKE, i colori di fondo per ciascuna area nelle locazioni 829 e 830, rispettivamente. Potete persino scegliere a quale riga di schermo volete che avvenga la suddivisione: limitatevi ad una istruzione POKE 828,50+(riga)*8.

Ecco le istruzioni DATA per l'interruzione di cursore schermo:

Programma 3. Raster interrupt

```
49020 CK=0:FORI=49236TO49331:READA:CK=CK+A
      :POKEI,A:NEXT:IFCK=10328THENRETURN
49030 PRINT"{RVS}ERRORE NELLE FRASI DATA:
      CONTROLLALE":STOP
```

```
49236 DATA120,169,127,141,013,220
49242 DATA169,001,141,026,208,173
49248 DATA060,003,141,018,208,169
49254 DATA027,141,017,208,169,118
49260 DATA141,020,003,169,192,141
49266 DATA021,003,088,096,173,018
49272 DATA208,205,060,003,208,028
49278 DATA169,000,141,018,208,173
49284 DATA064,003,141,024,208,173
49290 DATA062,003,141,033,208,169
49296 DATA001,141,025,208,104,168
49302 DATA104,170,104,064,173,060
49308 DATA003,141,018,208,173,061
49314 DATA003,141,033,208,173,063
49320 DATA003,141,024,208,169,001
49326 DATA141,025,208,076,049,234
```

Più sotto è riportata una riga che inizializza il sottoprogramma ed inserisce alcuni valori numerici esemplificativi. La parte superiore dello schermo è nera, la parte inferiore bianca. La metà superiore è in maiuscolo, la metà inferiore in minuscolo. La suddivisione è predisposta perché si verifichi alla ventesima riga (12*8+50=146).

```
10 GOSUB 49020:POKE 828,146:
   POKE 829,0:POKE 830,1:POK
   E 831,21:POKE 832,23:END
```

Non mandate in esecuzione «Copia-ROM» (con un comando SYS 49152) mentre il sottoprogramma di interruzione cursore di schermo è abilitato; prima di farlo, disabilitatelo. Dovrete disabilitarlo (con un comando POKE 53274,0) anche prima di una operazione di ingresso/uscita da nastro, quindi lo potete riabilitare con una SYS 49236.

Viene anche fornito il listato (Programma 4) del codice sorgente per entrambi i Programmi 1 e 3.

Programma 4. Codice sorgente

```
110:    C000                          .OPT  P4
120:    C000                          *=    $C000
130:    C000              SRC         =     $FB
140:    C000              DEST        =     $FD
150:    C000              TEMP        =     $A7
                          ;
170:    C000 78           COPYDOWN    SEI
170:    C001 AD 0E DC                 LDA   56334
170:    C004 29 FE                    AND   #254
170:    C006 8D 0E DC                 STA   56334
170:    C009 A5 01                    LDA   1
170:    C00B 29 FB                    AND   #251
170:    C00D 85 01                    STA   1
180:    C00F AD 18 D0                 LDA   53272
180:    C012 29 0E                    AND   #%1110
180:    C014 0A                       ASL
180:    C015 0A                       ASL
180:    C016 85 A7                    STA   TEMP
180:    C018 A9 D0                    LDA   #$D0
180:    C01A 85 FC                    STA   SRC+1
190:    C01C AD 00 DD                 LDA   56576
190:    C01F 29 03                    AND   #%11
190:    C021 49 03                    EOR   #%11
190:    C023 0A                       ASL
190:    C024 0A                       ASL
190:    C025 0A                       ASL
190:    C026 0A                       ASL
190:    C027 0A                       ASL
190:    C028 0A                       ASL
190:    C029 05 A7                    ORA   TEMP
190:    C02B 85 FE                    STA   DEST+1
200:    C02D A9 00                    LDA   #0
200:    C02F 85 FB                    STA   SRC
200:    C031 85 FD                    STA   DEST
200:    C033 A8                       TAY
200:    C034 A2 08                    LDX   #8
210:    C036 B1 FB        INLOOP      LDA   (SRC),Y
210:    C038 91 FD                    STA   (DEST),Y
210:    C03A C8                       INY
210:    C03B D0 F9                    BNE   INLOOP
220:    C03D E6 FC                    INC   SRC+1
220:    C03F E6 FE                    INC   DEST+1
220:    C041 CA                       DEX
220:    C042 D0 F2                    BNE   INLOOP
```

```
230:    C044 A5 01                  LDA   1
230:    C046 09 04                  ORA   #4
230:    C048 85 01                  STA   1
230:    C04A AD 0E DC               LDA   56334
230:    C04D 09 01                  ORA   #1
230:    C04F 8D 0E DC               STA   56334
230:    C052 58                     CLI
230:    C053 60                     RTS
                          ;
250:    C054 78           RASTER    SEI
250:    C055 A9 7F                  LDA   #$7F
250:    C057 8D 0D DC               STA   $DC0D
250:    C05A A9 01                  LDA   #1
250:    C05C 8D 1A D0               STA   $D01A
260:    C05F AD 3C 03               LDA   FIRST
260:    C062 8D 12 D0               STA   $D012
260:    C065 A9 1B                  LDA   #27
260:    C067 8D 11 D0               STA   $D011
270:    C06A A9 76                  LDA   #<IRQ
270:    C06C 8D 14 03               STA   $314
270:    C06F A9 C0                  LDA   #>IRQ
270:    C071 8D 15 03               STA   $315
270:    C074 58                     CLI
270:    C075 60                     RTS
                          ;
290:    C076 AD 12 D0     IRQ       LDA   $D012
290:    C079 CD 3C 03               CMP   FIRST
290:    C07C D0 1C                  BNE   TWO
300:    C07E A9 00                  LDA   #0
300:    C080 8D 12 D0               STA   $D012
300:    C083 AD 40 03               LDA   SHAD2
300:    C086 8D 18 D0               STA   53272
310:    C089 AD 3E 03               LDA   COL2
310:    C08C 8D 21 D0               STA   53281
310:    C08F A9 01                  LDA   #1
310:    C091 8D 19 D0               STA   $D019
320:    C094 68           IREXIT    PLA
320:    C095 A8                     TAY
320:    C096 68                     PLA
320:    C097 AA                     TAX
320:    C098 68                     PLA
320:    C099 40                     RTI
                          ;
340:    C09A AD 3C 03     TWO       LDA   FIRST
340:    C09D 8D 12 D0               STA   $D012
340:    C0A0 AD 3D 03               LDA   COL1
```

```
340:    C0A3 8D 21 D0             STA   53281
340:    C0A6 AD 3F 03             LDA   SHAD1
340:    C0A9 8D 18 D0             STA   53272
350:    C0AC A9 01               LDA   #1
350:    C0AE 8D 19 D0            STA   $D019
350:    C0B1 4C 31 EA            JMP   $EA31
                          ;
370:    C0B4              FIRST    =    828
380:    C0B4              COL1     =    829
390:    C0B4              COL2     =    830
400:    C0B4              SHAD1    =    831
410:    C0B4              SHAD2    =    832
```

Capitolo due

Azione

Costruiamo gli sprite

Stephen Meirowsky

Come creare e modificare semplicemente sprite multicolori con il Commodore 64.

Possibilità grafiche

Il C64 ha un formato grafico 40x25, proprio come il PET. In più ha la possibilità di usare gli *sprite* con la grafica in formato di testo. Questi strumenti vi permettono di disegnare figure personalizzate in quattro differenti colori (il manuale mostra come servirsi di un solo colore), proprio come i videogames delle sale giochi. Gli sprite possono avere uno dei 16 colori disponibili in formato monocolore e quattro dei primi otto colori in formato multicolore.

Sono disponibili, per essere visualizzati sullo schermo, otto sprite in un formato di 24 pixel orizzontali per 21 pixel verticali. Ogni sprite ha un differente grado gerarchico nell'incrociarsi con altri sprite. Lo sprite 0 passerà davanti allo sprite 1; lo sprite 1 e lo sprite 0 passeranno davanti allo sprite 2 e così via fino allo sprite 7. Tutti gli altri sprite passeranno davanti allo sprite 7. Inoltre potete segnalare ad ogni sprite se deve passare davanti o dietro la grafica di fondo in formato testo normale.

Ogni sprite può essere espanso al doppio delle dimensioni originali, orizzontalmente, verticalmente o in entrambe le direzioni. Un meccanismo di rilevazione automatica delle collisioni vi segnala quando più sprite si sono scontrati o quando uno sprite ha colpito la grafica di fondo.

Sul manuale del Commodore si trova il numero del registro, appartenente al chip grafico, che contiene informazioni sulle collisioni. Innanzi tutto il registro delle collisioni tra sprite è contraddistinto dal valore numerico decimale 30. Quando gli sprite si scontrano, il chip grafico pone a 1 i bit di questo registro corrispondenti agli sprite scontratisi. In secondo luogo, la collisione tra sprite

e grafica di fondo è segnalata dal registro 31. Quando uno sprite urta il fondo il suo bit corrispondente viene posto a 1.

Come si crea uno sprite

Per costruire uno sprite dovete innanzi tutto disegnarlo in una griglia 24x21. Quindi convertite i punti campiti in ciascuna riga in tre byte differenti, usando il codice binario. Per ciascun byte sommate il numero corrispondente ai suoi bit. I numeri per ogni bit, internamente ad un byte, sono 128, 64, 32, 16, 8, 4, 2 e 1.

Un esempio di conversione della griglia:

Riga 1 +......+.......++++++++
Riga 2 +..+...+.......++++++++
Riga 3 ...+...+.......+++. . .+++

```
101 DATA 129,1,255:REM DATI DELLA RIGA 1
102 DATA 145,1,255:REM DATI DELLA RIGA 2
103 DATA  17,1,199:REM DATI DELLA RIGA 3
104 DATA
```

In seguito inserite, tramite una istruzione POKE, i 63 byte di dati in memoria per descrivere gli sprite al calcolatore. La conversione della griglia in 63 byte non è difficile, ma richiede parecchio tempo. Ecco perché abbiamo realizzato questo programma.

Il sistema più comodo

«Costruiamo gli sprite» vi fornisce parecchi comandi semplici, ad un solo tasto, per costruire gli sprite, visualizzarli e registrarli. Quando il programma viene mandato in esecuzione i comandi vengono visualizzati lungo il lato sinistro dello schermo. Sul lato destro dello schermo c'è una griglia 24x21 che viene usata per costruire gli sprite. Per spostare il cursore usate i tasti cursore. Se volete che un pixel venga attivato sullo sprite, premete i tasti 1, 2 o 3. Se volete che il pixel venga cancellato, premete il tasto "←" (frecciolina a sinistra). Tutte le volte che volete vedere la forma attuale dello sprite, premete il tasto "=", che calcolerà la configurazione di byte della griglia e visualizzerà lo sprite nell'angolo inferiore sinistro.

Se operate un qualsiasi cambiamento sulla griglia, lo sprite non verrà visualizzato nell'angolo fino a che verrà premuto nuovamente il tasto "=". Una volta che lo sprite è stato visualizzato può essere

espanso orizzontalmente o verticalmente, premendo X o Y. Inoltre, premendo B, potete anche visualizzare i dati necessari per utilizzare questo sprite in un programma.

Per tutti e quattro i comandi seguenti il calcolatore opererà un controllo di correttezza, chiedendo se è effettivamente il comando che volete venga eseguito. I quattro comandi sono: N per cancellare la griglia e lo sprite, in modo da crearne uno nuovo; S per registrare i dati dello sprite su nastro; L per leggere uno sprite da nastro e F per abbandonare il programma.

È conveniente calcolare i dati dello sprite (premendo "=") prima di visualizzarli (premendo "B") e registrarli (premendo "S"), per assicurarsi che lo sprite sia stato aggiornato.

Per cambiare colori mentre si creano gli sprite usate i tasti f1, f3, f5 e f7.

Disco o nastro

Il programma, così come è stato scritto, è predisposto per l'uso dell'unità nastro. Per registrare uno sprite su disco è necessario cambiare le righe 196 e 200 nel modo indicato dalle istruzioni REM di riga 196 e riga 201.

Costruiamo gli sprite

```
10 POKE53281,6:DIM A(21,24),B(63),A$(15) :
   X=0:Y=0:R=0:C=0:S=1039:S1=55311
11 V=53248:POKEV+21,0:POKEV+23,0:POKEV+29,
   0:RESTORE:FORX=0TO15:READA$(X):NEXT
12 PRINT"{CLR}":FORR=1 TO 21:FOR C=1 TO 24
   :A(R,C)=46:NEXT:NEXT
13 FOR X=1 TO 63:B(X)=0:NEXT
14 POKEV+4,60:POKEV+5,200:POKE2042,13:POKE
   V+37,0:POKEV+41,14:POKEV+38,1
16 FORX=1TO63:POKE831+X,B(X):NEXT:POKEV+21
   ,4:POKE V+28,4
20 PRINT"{CLR}{GIU'}[<7>]MC SPRITE EDITOR
   {GIU'}"
22 PRINT"← CANCELLA"
23 PRINT"1 MC 0-"A$(PEEK(V+37) AND 15)
24 PRINT"2 SC{ 2 SPAZI}-"A$(PEEK(V+41) AND
    15)
```

```
25 PRINT"3 MC 1-"A$(PEEK(V+38) AND 15)
32 PRINT"= CALCOLA I DATI"
33 PRINT"X SCALA 'X'"
34 PRINT"Y SCALA 'Y'"
35 PRINT"B DATI"
36 PRINT"N CANCELLA"
37 PRINT"S SALVA"
38 PRINT"L CARICA"
39 PRINT"F FINE{GIU'}"
50 Y=0:FORR=1TO21:FORC=1TO24:Y=Y+1:POKES+Y
   ,A(R,C):POKES1+Y,14:NEXT:Y=Y+16:NEXT
55 X=1:Y=1:GOTO79
60 GETA$:IFA$=""THEN60
61 R=S+X+(Y-1)*40:C=A(Y,X):POKER,C:POKER+1
   ,C
62 IFA$="{GIU'}"THENY=Y+1:IFY>21THENY=1
63 IFA$="{SU}"THENY=Y-1:IFY<1THENY=21
64 IFA$="{DES}"THENX=X+2:IFX>24THENX=1
65 IFA$="{SIN}"THENX=X-2:IFX<1THENX=23
66 IFA$="←"THENA(Y,X)=46:A(X,Y+1)=46
67 IFA$>"0"AND A$<"4"THENR=48+VAL(A$):A(Y,
   X)=R:A(Y,X+1)=R
68 IFA$="="THEN100
69 IFA$="X"THENPOKEV+29,ABS(PEEK(V+29)-4)
70 IFA$="Y"THENPOKEV+23,ABS(PEEK(V+23)-4)
71 IFA$="B"THEN120
72 IFA$="L"ORA$="S"ORA$="N"ORA$="F"THEN190

73 IF A$="{F1}"THENR=33:GOSUB 130
74 IF A$="{F3}"THENR=37:GOSUB 130
75 IF A$="{F5}"THENR=41:GOSUB 130
76 IF A$="{F7}"THENR=38:GOSUB 130
79 R=S+X+(Y-1)*40:C=A(Y,X)+128:POKER,C:POK
   ER+1,C:GOTO60
100 Y=0:FORR=1TO21:FORX=0TO2:Y=Y+1:B(Y)=0:
    FORC=1TO7STEP2:Q=A(R,X*8+C)-48
102 IFQ<0 OR Q>3 THEN Q=0
104 B(Y)=B(Y)+2↑(7-C)*Q:NEXT:NEXT:NEXT:FOR
    X=1TO63:POKE831+X,B(X):NEXT:GOTO55
```

```
110 PRINT"{RVS}"A$": SI O NO"
111 FORX=1TO10:GETN$:NEXT
112 GETN$:IFN$=""THEN112
114 PRINT"{SU}{ 16 SPAZI}{SU}":RETURN
115 PRINT"{RVS}CONTINUAZIONE":GOTO111
119 REM
120 PRINT"{CLR}":FORX=1TO7:PRINT"DATA";:FO
    RY=1TO9:PRINTB((X-1)*9+Y)"{SIN},";:NEX
    T
122 PRINT"{SIN} ":NEXT:PRINT:GOSUB115:GOTO
    20
130 C=PEEK(V+R)AND15:C=C+1:IF C>15 THEN C=
    0
132 POKE V+R,C:PRINT"{HOME}{ 3 GIU'}";:IFR
    =33 THEN 136
133 PRINT"{GIU'}";:IF R=37 THEN 136
134 PRINT"{GIU'}";:IF R=41 THEN 136
135 PRINT"{GIU'}";
136 PRINT"{ 7 DES}"A$(C)"{ 2 SPAZI}":RETUR
    N
190 GOSUB110:IFN$<>"S"THEN79
191 GETN$:GETN$:IFA$="N"THEN11
192 IFA$="Q"THENPOKEV+21,0:PRINT"
    { 4 GIU'}":END
194 PRINT"{CLR}":POKEV+21,0:INPUT"NOME DEL
    LO SPRITE";N$:PRINT
196 IFA$="L"THENOPEN1,1,0,N$:GOTO300:REM P
    ER I DISCHI METTERE OPEN 1,8,2,N$
200 OPEN1,1,1,N$:FORX=1TO63:PRINT#1,B(X):N
    EXT:CLOSE1:GOTO16
201 REM DISK USERS OPEN 1,8,2,N$+",S,W" ON
     LINE 200
300 FORX=1TO63:INPUT#1,B(X):NEXT:CLOSE1:PR
    INT"{GIU'}COMPUTING SPRITE MATRIX"
301 PRINT" SPRITE"
310 Y=0:FORR=1TO21:FORX=0TO2:Y=Y+1:FORC=2T
    O8STEP2:Q=X*8+C:P=2↑(8-C)
312 S=B(Y)AND(P*3):A(R,Q)=46:A(R,Q-1)=46
314 IF S>0 THEN A(R,Q)=S/P+48:A(R,Q-1)=S/P
```

```
    +48
330 NEXT:NEXT:NEXT:S=1039:GOTO16
500 DATA NERO,BIANCO,ROSSO,BLU-V.,PORPORA,
    VERDE,BLU,GIALLO
510 DATA ARANCIO,MARRONE,ROSSO C,GRIGIO1,G
    RIGIO2,VERDE C,AZZURRO
520 DATAGRIGIO3
```

Come animare gli sprite

Eric Brandon

Viene presentata una trattazione dettagliata di come l'autore, usando gli sprite, è stato in grado di creare una realistica simulazione del decollo di una navetta spaziale. Vengono spiegate diverse, utili tecniche di animazione.

Dopo essere stato in Florida per assistere al decollo della navetta spaziale Challenger, scrissi un breve programma per far vedere agli amici lo spettacolo cui avevo assistito. Per pura coincidenza, quello stesso giorno stavo scrivendo una versione per Commodore 64 del gioco «Labirinto rotante» di Matt Giwer. Ecco come è nato «Shuttle in fuga».

Il programma che potete vedere in questo articolo è la pagina di presentazione di «Shuttle in fuga». È essenzialmente il mio programma originale, ripulito un po' per gli scopi del gioco. Nel programma vengono utilizzate parecchie tecniche interessanti, che possono essere facilmente adattate a qualsiasi esempio di programmazione delle animazioni.

Gli aspetti più importanti innanzi tutto

La prima cosa che il programma fa è richiamare con una istruzione GOSUB la riga 3000, dove esegue un sottoprogramma che stampa le parole SHUTTLE IN FUGA sullo schermo, a titoli di scatola. Quindi, in riga 110, stampa un CHRS(142) per assicurarsi che lo schermo sia in formato grafico e non in formato testo maiuscolo/minuscolo.

La riga 120 controlla se i dati dello sprite sono già in memoria. Se non lo sono, richiama un sottoprogramma in riga 10000 che inserisce i dati dello sprite tramite istruzioni POKE.

Le ragioni per cui sono necessari tutti quei numeri

Le righe da 10000 a 10026 rappresentano un semplice ciclo di lettura dei valori delle istruzioni DATA, tramite READ, ed inseri-

mento in memoria con delle istruzioni POKE. In riga 10000 l'orologio interno, la variabile TI$, viene azzerato. In riga 10005 un conteggio alla rovescia di 43 secondi viene mostrato sottraendo TI/60 dal numero 43 e visualizzandolo al centro del messaggio. La variabile TI$ contiene il tempo nel formato OOMMSS (ore, minuti, secondi), ma la variabile TI dà lo stesso tempo in sessantesimi di secondo, chiamati jiffies. Vi potreste chiedere come facessi a sapere a priori che il conto alla rovescia sarebbe durato 43 secondi. La risposta è che non lo sapevo. Ho provato valori diversi fino a che ho trovato quello che ha fatto terminare il conteggio alla rovescia con 0 secondi sull'orologio. La maggior parte di questo tipo di programmazione consiste nell'aggiustare i dati fino a che hanno un aspetto convincente. La somma di controllo in riga 10025 è stata ottenuta scrivendo un breve programma per leggere, con una istruzione READ, e sommare il contenuto delle istruzioni DATA. La ragione per cui è stata introdotta la somma di controllo è di farvi sapere se avete commesso degli errori di copiatura.

In riga 10030 iniziano le istruzioni DATA che contengono la forma degli sprite. La forma di ogni sprite consiste di 64 valori numerici. Ogni sprite ha una larghezza di 24 pixel ed un'altezza di 21, per un totale di 504 punti. Ciascun punto è rappresentato da un bit; 504 bit diviso 8 bit per byte = 63 byte. Un 64-esimo byte è necessario per sistemare la definizione degli sprite. Le forme, nell'ordine in cui appaiono nelle istruzioni DATA, sono:

Pagina	**Forma**
244	Shuttle in posizione orizzontale.
245	Shuttle in posizione verticale con la parte inferiore del serbatoio del carburante.
246	Ogiva del serbatoio del carburante.
247	Fiammata di piccole dimensioni.
248	Fiammata di dimensioni maggiori.
249	Shuttle in posizione verticale senza serbatoio del carburante.
250	Parte inferiore del serbatoio del carburante (senza la navetta).
251	Serbatoio del carburante parzialmente disintegrato.
252	Serbatoio del carburante ulteriormente disintegrato.
253	Ogiva del serbatoio del carburante parzialmente disintegrata.

254 Forme diverse, delle stesse dimensioni della pàgina 248.

Che cosa si intende per *pagina*? Ogni sprite ha un puntatore alle locazioni di memoria 2040-2047 che segnala dove trovare i dati che compongono la sua forma. Il valore numerico che inserite in questo puntatore è il suo numero di *pagina*. Ciascuna pagina ha una lunghezza di 64 byte, quindi la pagina 244 inizia alla locazione di memoria 244*64, cioè 15616. Le ragiôni per cui sono necessari così tanti sprite differenti diventeranno evidenti mentre ci addentreremo nei meandri del programma.

Ed ora, che me ne faccio di questi sprite?

In riga 130 la variabile V è inizializzata a 13*4096 ($D000 o 53248). V punta al chip video VIC-II, dove vengono controllate tutte le funzioni dello schermo. La variabile CO (colonna) viene inizializzata a 50. La funzione della variabile CO è di spostare tutte le posizioni X dei vari sprite di 50 pixel. Inserendo questo valore numerico in una variabile, posso spostare la sequenza di decollo a destra o a sinistra sullo schermo, quanto è necessario, quando vengono aggiunte altre parti della pagina di presentazione.

A questo punto dovevo decidere, per ogni immagine, quale sprite dovesse visualizzarla. Scelsi arbitrariamente lo sprite 0 per la navetta spaziale, 1 per il serbatoio del carburante, 2 per la fiammata e 3 per l'ogiva del serbatoio del carburante. La ragione per cui l'ogiva segue la fiammata è data dal fatto che avevo già posto una fiammata nel disegno prima di decidere di prolungare il tozzo serbatoio del carburante.

Questo ci porta alla riga 140, dove comincia un'orgia di POKE. L'istruzione POKE V+16,0 cancella il bit 9 della ascissa X di tutti gli sprite. In questo modo possiamo essere sicuri che tutte le fasi del decollo avvengano omogeneamente sulla parte destra dello schermo. Per avere un'idea più precisa di ciò che questo significa provate a cambiare questa riga, inserendo valori numerici diversi da 0 nella locazione V+16, tramite istruzioni POKE.

La riga 160 inizializza l'ascissa X della navetta, ed il serbatoio del carburante ad essa connesso, al valore numerico contenuto nella variabile CO. La coordinata della fiammata alla base della navetta viene inizializzata a CO-2. L'istruzione POKE V+5,221 inizializza la ordinata della fiammata. La riga successiva inizializza la ordinata Y

del complesso navetta/serbatoio in 200. La fiammata è in 221 e 221-200=21, cioè l'altezza dello sprite. Molti degli sprite di questo programma sono separati nella direzione Y da 21 pixel, per la stessa ragione.

Le righe 180 e 190 pongono l'ogiva del serbatoio del carburante proprio sopra la navetta. CO rappresenta la sua ascissa X e la sua ordinata Y è 200-21=179.

La riga 210 inizializza il colore degli sprite 0, 1 e 3 al bianco, colore 1. Pone anche il colore dello sprite 2 ad arancio, colore 8.

Nelle righe da 220 a 240 i valori numerici di pagina vengono inseriti, tramite una istruzione POKE, nei puntatori ai dati degli sprite. Notate che i valori numerici delle pagine corrispondono alla tavola precedente.

In riga 250 abbiamo un breve ritardo e quindi, con l'istruzione POKE V+21,7, poniamo tre sprite sullo schermo. Il contenuto di V+21 comunica al chip VIC-II quale sprite mostrare. Tutte le volte che un bit di questo byte viene posto a uno apparirà uno sprite. Dal momento che 7 è in binario 00000111, questa istruzione POKE attiva gli sprite 0, 1 e 2.

Un po' di rumore

La riga 260 ci rinvia al sottoprogramma sonoro di riga 2000. In questo sottoprogramma la variabile S viene inizializzata a 54272. S contiene l'indirizzo di base del chip che gestisce gli effetti sonori. In riga 2010 il volume viene posto al massimo e vengono attivati sia il filtro passa-alto che il filtro passa-basso. Ciò significa che le frequenze più alte e più basse passano inalterate attraverso il chip SID (chip sonoro), ma che la parte centrale della banda viene smorzata. Come si attivano questi filtri? Semplice. L'istruzione POKE S+24,15 mette al massimo il volume e non fa nient'altro. Aggiungendo 16 si pone a 1 il bit 4 ed aggiungendo 32 si pone a 1 il bit 5. Questi due bit controllano i filtri.

La successiva POKE, alla locazione S+23, ha due effetti. Ponendo a 1 il bit 0, invia l'uscita della voce 1 attraverso i filtri. Inserendo un 5 nei 4 bit più significativi, pone il valore di risonanza a 5. La risonanza determina l'acutezza del suono al punto di taglio dei filtri.

L'attaccare ed il rilasciare vengono posti a 0 in riga 2020, in modo da ottenere un suono che comincia immediatamente. La riga 2030 pone sostenere e decadere ai valori massimi. In riga 2040 viene

attivata la forma d'onda del rumore ed infine in riga 2050 il byte più significativo della frequenza viene inizializzato a 11.

Il suono rappresenta senza dubbio la parte più complicata di un programma come questo, ma attraverso una serie di prove intelligenti e di errori dovreste essere in grado di ottenere dei buoni effetti sonori. Ad esempio, se non avete idea di quale effetto provocherebbe un filtro sul suono che state progettando di ottenere, attivatelo e sentite. Se il risultato non vi soddisfa, potete sempre rinunciarvi.

Accensione!

La variabile I rappresenta la ordinata della navetta e viene inizializzata in riga 270. La navetta spaziale accelera non appena si stacca dal suolo; quindi l'ordinata della navetta deve rapidamente sottrarre numeri sempre più grandi. La prima cosa che viene in mente è di usare l'equazione di una parabola (la velocità è proporzionale al quadrato del numero di volte in cui il ciclo è stato compiuto). Quando provai con questa equazione la navetta non sembrava accelerare abbastanza rapidamente, quindi ho aggiunto la variabile P, un terzo coefficiente. Tutto ciò significa che la variabile Q rappresenta la velocità di movimento della navetta. Questa velocità viene aumentata aggiungendo ripetutamente .01*P, e P viene aumentato in modo da simulare la rapida diminuzione di peso della navetta quando consuma il suo carburante.

La variabile C in riga 300 è un contatore. Ci dice fino a che punto è arrivato il programma e quando dare inizio ai vari passi del decollo.

Le righe 320 e 330 alternano le due immagini leggermente differenti delle grandi fiammate a pagina 248 ed a pagina 254. Dal momento che all'inizio del programma la fiammata è rappresentata dalla immagine più piccola di pagina 247, queste righe non entrano in funzione che più avanti nel corso del programma. Tuttavia servono ad illustrare un aspetto importante dell'animazione degli sprite. Cambiando l'indirizzo identificato dal puntatore degli sprite, potete istantaneamente, e praticamente senza alcun ulteriore sovraccarico del programma, cambiare la forma di un oggetto sullo schermo. Questo meccanismo viene usato ripetutamente in questo programma ed in ogni programma di questo tipo.

La riga 340 inizializza l'ordinata della navetta, dell'ogiva del serbatoio del carburante e della fiammata. Sono tutte correlate alla

stessa variabile, I, così che cambiando il valore di una sola variabile, potete cambiare l'ordinata di tutti e tre gli sprite.

Le righe da 350 a 360 cambiano dinamicamente la frequenza di taglio del filtro. Ciò fornisce al suono una specie di rombo, molto più realistico di un ronzio continuo. In riga 360 la variabile P2, la frequenza di taglio, viene aumentata.

Le righe 370-390 controllano la fiammata. Il registro V+23 controlla se gli sprite vengono espansi nella direzione Y. Inserendo un 4 in questo registro, raddoppiamo la lunghezza in verticale dello sprite 2, la fiammata. La prima riga ad essere eseguita è la 380, quando il contatore raggiunge il valore numerico di 20. Questa riga raddoppia le dimensioni del piccolo sprite. La riga successiva, 390, verrà eseguita quando il contatore raggiunge 40. Questa riga disattiva l'espansione nel verso Y, ma cambia il puntatore agli sprite ad una delle fiammate più grandi. Con pochissimo disturbo siamo riusciti a visualizzare sei differenti tipi di fiammata.

La riga 400 mantiene il ciclo in esecuzione fino a che il contatore è inferiore a 70.

Primo stadio!

Dopo che il contatore ha raggiunto 70, è tempo che il serbatoio del carburante si stacchi. A questo punto alcuni di voi dovrebbero domandarsi dove siano i razzi propulsori. Non sono stati inseriti perché il decollo sembrava abbastanza ben riuscito così.

Se il serbatoio e la navetta devono separarsi, devono diventare due sprite differenti. Ecco perché, nelle righe 410 e 420, la navetta assume l'aspetto di pagina 249 ed il serbatoio quello di pagina 250. Le coordinate X e Y del serbatoio (sprite 3) vengono inizializzate e quindi, con una istruzione POKE V+21,15, aggiungiamo lo sprite 3 agli altri già visibili.

Le righe da 430 a 470 sono delle copie esatte delle righe da 290 a 330 e in questo nuovo ciclo servono esattamente allo stesso scopo. La sola differenza consiste nell'introduzione della nuova variabile C2. Questa variabile, che viene aumentata di 0.6 ad ogni iterazione del ciclo, viene usata per calcolare le coordinate del serbatoio del carburante in caduta libera.

La riga 490 calcola la posizione del serbatoio che cade. L'ordinata Y è contenuta nella variabile NR (nuova riga) e viene calcolata da I+C2*C2. La ragione di questa formula consiste nel fatto che quando la variabile C2 contiene un valore numerico piccolo, la variabile

I viene rapidamente decrementata ed il serbatoio del carburante si alza. Man mano che la variabile C2 viene incrementata, il suo effetto comincia a predominare ed il serbatoio cade. Questo crea l'illusione che il serbatoio venga portato verso l'alto dall'abbrivio e quindi attirato verso il basso, a velocità sempre maggiore, dalla forza di gravità. L'ascissa x del serbatoio, la variabile NC (nuova colonna), viene semplicemente calcolata come un multiplo del contatore C2 sommato alla variabile CO, la colonna di base.

La riga 500 inserisce, tramite istruzioni POKE, le coordinate X ed Y del serbatoio del carburante e dell'ogiva del serbatoio stesso.

Le righe da 510 a 530 controllano il disintegrarsi del serbatoio del carburante. Quando il Challenger sgancia il suo serbatoio del carburante, questo cade per alcuni chilometri e quindi brucia a contatto con l'atmosfera. Questo programma simula questo effetto, ponendo i numeri di pagina di serbatoi sempre più disintegrati nel puntatore per lo sprite 3. La riga 510 cambia i puntatori sia del serbatoio che dell'ogiva del serbatoio quando il contatore, la variabile C, raggiunge 83. Quando C è uguale a 86, la riga 520 cambia nuovamente il puntatore del serbatoio. L'ogiva non cambia ulteriormente, ma, dal momento che è piuttosto piccola, ciò non danneggia eccessivamente l'effetto. Infine, quando il contatore raggiunge 89, la riga 530 opera una istruzione POKE V+21,5, lasciando sullo schermo solo gli sprite 0 e 2 ed eliminando del tutto il serbatoio del carburante.

Il suono comincia a decadere con la riga 570; impiega 24 secondi a raggiungere il volume minimo. Questo fatto dà l'impressione che la navetta si stia allontanando in distanza anche quando non compare più sullo schermo.

Orbita raggiunta

Ora la navetta attraversa la scritta SHUTTLE IN FUGA come simbolo del fatto che l'orbita è stata raggiunta.

La riga 600 inserisce il valore numerico 1 nella locazione V+21 con una POKE, lasciando potenzialmente in vista solo lo sprite 0.

La riga 610 è un ciclo che dà al suono il tempo di smorzarsi.

La riga 620 stampa «Orbita raggiunta...» sulla parte superiore dello schermo per evitare confusione circa ciò che sta accadendo.

La riga 640 indirizza il puntatore agli sprite alla pagina 244, con-

tenente la navetta in posizione orizzontale.

La riga 650 pone la navetta al margine sinistro dello schermo, a 117 pixel dal margine superiore.

Le righe da 660 a 680 spostano la navetta, attraverso lo schermo, di due pixel per volta. In riga 670 dobbiamo preoccuparci per la prima volta del margine degli sprite della locazione di ascissa 255. Dal momento che ciascun byte può contenere solo un valore numerico compreso tra 0 e 255, mentre ci sono 320 pixel longitudinalmente attraverso lo schermo, l'ascissa X di ciascuno sprite deve essere contenuta in più di un byte. Nella locazione V+16, ogni sprite ha un bit che, quando viene posto a 1, calcola la sua ascissa X non a partire dal margine sinistro dello schermo, ma 256 pixel più a destra. La riga 670 si occupa di ciò inserendo semplicemente gli otto bit meno significativi della variabile I in V, con una istruzione POKE, ed inserendo il nono bit nella locazione V+16.

Ora tocca a voi

Benché il programma sia complicato, le tecniche utilizzate sono semplici e facilmente applicabili ai vostri programmi.

La cosa più importante da imparare da questo programma è che le animazioni non devono necessariamente seguire le effettive leggi della fisica per avere un aspetto realistico; le equazioni possono essere semplificate. Un altro aspetto importante consiste nel fatto che quasi tutti gli effetti non avranno un buon aspetto quando vengono programmati per la prima volta. Devono essere adattati, fino a quando non hanno l'aspetto desiderato.

N.B. — Inserire il listato del programma «Shuttle in fuga». Riferimento su disco 4/193/I.

Shuttle in fuga

```
100 GOSUB3000
110 PRINTCHR$(142)
120 IF PEEK(15625)<>24 THEN GOSUB 10000
130 V=13*4096:CO=50
140 POKE V+16,0
160 POKE V+0,CO:POKEV+4,CO-2:POKEV+5,221
170 POKE V+1,200
180 POKE V+2,CO
190 POKE V+3,179
```

```
210 POKE V+39,1:POKEV+40,1:POKEV+41,8:POKE
    V+42,1
220 POKE 2040,245
230 POKE 2041,246:POKE2043,246
240 POKE 2042,247
250 FOR K=1 TO 500 : NEXT K:POKEV+21,7
260 GOSUB 2000
270 I=200
280 P=1
290 Q=Q+.01*P
300 P=P+.1:C=C+1
310 I=I-Q
320 IF PEEK(2042)=248 THEN POKE 2042,254:G
    OTO340
330 IF PEEK(2042)=254 THEN POKE 2042,248
340 POKE V+1,I:POKEV+3,I-21:POKEV+5,I+21
350 POKES+22,P2:POKES+23,1OR(16-P2/16)*16
360 P2=P2+P2/244
370 IF C=60THEN POKEV+23,4
380 IF C=20THEN POKEV+23,4
390 IF C=40 THEN POKEV+23,0:POKE2042,248
400 IF C<70 THEN 290
410 POKE 2040,249
420 POKE 2043,250:POKEV+6,CO:POKEV+7,I:POK
    EV+21,15
430 Q=Q+.01*P
440 P=P+.1:C=C+1:C2=C2+.6
450 I=I-Q
460 IF PEEK(2042)=248 THEN POKE 2042,254:G
    OTO480
470 IF PEEK(2042)=254 THEN POKE 2042,248
480 POKE V+1,I:POKEV+5,I+21
490 NR=I+C2*C2:NC=CO+C2*3
500 POKE V+7,NR:POKEV+3,NR-21:POKEV+6,NC:P
    OKEV+2,NC
510 IF C=83 THEN POKE 2043,251:POKE2041,25
    3
520 IF C=86 THEN POKE 2043,252
530 IF C=89 THEN POKE V+21,5
```

```
540 POKES+22,P2:POKES+23,1OR(16-P2/16)*16
550 P2=P2+P2/244
560 IF I>25 THEN 430
570 POKE S+4,128
580 POKE V+5,I+21
590 I=I-2:IFI>0 THEN580
600 POKE V+21,1
610 FOR J=1 TO 2000:NEXT
620 PRINT"{HOME}{ 10 DES}{WHT}{ 2 SPAZI}OR
    BITA RAGGIUNTA...{ 3 SPAZI}"
630 FOR I=1 TO 1000:NEXT
640 POKE 2040,244
650 POKE V,0:POKEV+1,117
660 FOR I=0 TO 348 STEP2
670 POKE V,I AND 255:POKEV+16,I/255
680 NEXT
690 FOR I=0 TO 1000:NEXT
720 END
2000 S=54272
2010 POKES+24,15+16+32:POKES+23,1+16*5
2020 POKES+5,0
2030 POKES+6,16*15+15
2040 POKES+4,129
2050 POKES+1,11
2060 P2=100:RETURN
3000 POKE 53281,0:POKE53280,0
3010 PRINT"{CLR}"
3020 PRINT"{ 5 GIU'}"
3040 T=12
3050 PRINTTAB(T)"[<7>]{RVS}£{ 2 SPAZI}
     {DES} {DES} {DES} {DES} {DES}
     { 3 SPAZI}{DES}{ 3 SPAZI}{DES}
     { 3 DES}£{ 2 SPAZI}"
3060 PRINTTAB(T)"{RVS} { 3 DES} {DES}
     {DES} {DES} { 2 DES} { 3 DES}
     { 2 DES} { 3 DES} "
3070 PRINTTAB(T)"[<*>]{RVS} [<*>]{DES}
     { 3 SPAZI}{DES} {DES} { 2 DES}
     { 3 DES} { 2 DES} { 3 DES}{ 2 SPAZI}
```

```
    "
3080 PRINTTAB(T)"{RVS}{ 2 DES} {DES} {DES}
      {DES} {DES} { 2 DES} { 3 DES}
     { 2 DES} { 3 DES} "
3090 PRINTTAB(T)"{RVS}{ 2 SPAZI}{OFF}£
     {RVS}{DES} {DES} {DES}{OFF}[<*>]{RVS}
      {OFF}£{RVS}{ 2 DES} { 3 DES}
     { 2 DES}{OFF}[<*>]{RVS}{ 2 SPAZI}
     {DES}{OFF}[<*>]{RVS}{ 2 SPAZI}"
3100 PRINT
3110 PRINTTAB(T)"{CYN} {RVS} {OFF}
     { 2 SPAZI}{RVS} {OFF} {RVS} {OFF}
     { 5 SPAZI}{RVS}£{ 2 SPAZI}{OFF} {RVS}
      {OFF} {RVS} {OFF} {RVS}£ [<*>]{OFF}
     {RVS}£ [<*>]{OFF}"
3120 PRINTTAB(T)" {RVS} {OFF}{ 2 SPAZI}
     {RVS} [<*>] {OFF}{ 5 SPAZI}{RVS}
     {OFF}{ 3 SPAZI}{RVS} {OFF} {RVS}
     {OFF} {RVS} {OFF} {RVS} {OFF} {RVS}
     {OFF} {RVS} {OFF}"
3130 PRINTTAB(T)" {RVS} {OFF}{ 2 SPAZI}
     {RVS} {OFF}[<*>]{RVS} {OFF}
     { 5 SPAZI}{RVS}{ 2 SPAZI}{OFF}
     { 2 SPAZI}{RVS} {OFF} {RVS} {OFF}
     {RVS} {OFF}{ 3 SPAZI}{RVS}{ 3 SPAZI}
     {OFF}"
3140 PRINTTAB(T)" {RVS} {OFF}{ 2 SPAZI}
     {RVS} {OFF} {RVS} {OFF}{ 5 SPAZI}
     {RVS} {OFF}{ 3 SPAZI}{RVS} {OFF}
     {RVS} {OFF} {RVS} {OFF}[<C>]{RVS}
     {OFF} {RVS} {OFF} {RVS} {OFF}"
3150 PRINTTAB(T)" {RVS} {OFF}{ 2 SPAZI}
     {RVS} {OFF} {RVS} {OFF}{ 5 SPAZI}
     {RVS} {OFF}{ 3 SPAZI}[<*>]{RVS} {OFF}
     £ [<*>]{RVS} {OFF}£ {RVS} {OFF} {RVS}
      {OFF}"
3999 RETURN
10000 I=15616:TI$="000000"
10005 PRINT"{HOME}{WHT}{ 12 DES}PRONTO IN"
```

```
      LEFT$(STR$(43-INT(TI/60)),4)" SECOND
      I "
10010 READ A:IF A=256 THEN RETURN
10020 C1=C1+A:POKE I,A:I=I+1:GOTO 10005
10025 IFC1<>30584 THEN PRINT"ERRORE NELLA
      SOMMA DI CONTROLLO DI RIGA 10025":EN
      D
10030 DATA 0,0,0,0,0,0,0
10040 DATA 0,0,24,0,0,28,0
10050 DATA 0,31,0,0,31,255,240
10060 DATA 31,255,8,20,255,254,31
10070 DATA 127,255,30,63,254,24,0
10080 DATA 0,0,0,0,0,0,0
10090 DATA 0,0,0,0,0,0,0
10100 DATA 0,0,0,0,0,0,0
10110 DATA 0,0,0,0,0,0,0
10120 DATA 0,0,71,192,0,247,192
10130 DATA 0,247,192,1,255,192,2
10140 DATA 255,192,2,255,192,2,247
10150 DATA 192,2,247,192,3,247,192
10160 DATA 3,247,192,3,247,192,3
10170 DATA 247,192,3,247,192,3,247
10180 DATA 192,3,255,192,3,255,192
10190 DATA 7,103,192,7,103,192,15
10200 DATA 229,128,31,119,128,31,240
10210 DATA 0,0,0,0,0,0,0
10220 DATA 0,0,0,0,0,0,0
10230 DATA 0,0,0,0,0,0,0
10240 DATA 0,0,0,0,0,0,0
10250 DATA 0,0,0,0,0,0,0
10260 DATA 0,0,0,0,0,0,0
10270 DATA 0,0,0,0,0,0,0
10280 DATA 0,0,3,128,0,15,192
10290 DATA 0,15,192,0,15,192,0
10300 DATA 15,192,0,1,252,0,1
10310 DATA 116,0,1,212,0,0,88
10320 DATA 0,0,80,0,0,0,0
10330 DATA 0,0,0,0,0,0,0
10340 DATA 0,0,0,0,0,0,0
```

```
10350 DATA 0,0,0,0,0,0,0
10360 DATA 0,0,0,0,0,0,0
10370 DATA 0,0,0,0,0,0,0
10380 DATA 0,0,0,0,0,0,0
10390 DATA 0,0,0,0,1,252,0
10400 DATA 1,252,0,1,252,0,1
10410 DATA 254,0,7,248,0,6,249
10420 DATA 0,2,251,0,6,122,0
10430 DATA 3,242,0,0,248,0,0
10440 DATA 248,0,0,60,0,0,120
10450 DATA 0,0,56,0,0,56,0
10460 DATA 0,96,0,0,96,0,0
10470 DATA 8,0,0,32,0,0,0
10480 DATA 0,0,0,0,0,0,64
10490 DATA 0,0,240,0,0,240,0
10500 DATA 1,240,0,2,240,0,2
10510 DATA 240,0,2,240,0,2,240
10520 DATA 0,3,240,0,3,240,0
10530 DATA 3,240,0,3,240,0,3
10540 DATA 240,0,3,240,0,3,240
10550 DATA 0,3,240,0,7,96,0
10560 DATA 7,96,0,15,224,0,31
10570 DATA 112,0,31,240,0,0,0
10580 DATA 7,192,0,7,192,0,7
10590 DATA 192,0,7,192,0,7,192
10600 DATA 0,7,192,0,7,192,0
10610 DATA 7,192,0,7,192,0,7
10620 DATA 192,0,7,192,0,7,192
10630 DATA 0,7,192,0,7,192,0
10640 DATA 7,192,0,7,192,0,7
10650 DATA 192,0,7,192,0,7,192
10660 DATA 0,3,128,0,0,0,0
10670 DATA 0,2,0,0,7,192,0
10680 DATA 7,192,0,6,192,0,4
10690 DATA 192,0,3,64,0,6,192
10700 DATA 0,1,192,0,4,0,0
10710 DATA 7,192,0,7,128,0,7
10720 DATA 64,0,7,192,0,1,192
10730 DATA 0,5,192,0,6,64,0
```

```
10740 DATA 7,192,0,7,192,0,0
10750 DATA 128,0,3,128,0,0,0
10760 DATA 0,0,2,0,0,1,0
10770 DATA 0,6,64,0,0,64,0
10780 DATA 4,128,0,3,64,0,6
10790 DATA 0,0,1,0,0,0,0
10800 DATA 0,0,0,0,0,128,0
10810 DATA 1,64,0,6,0,0,1
10820 DATA 0,0,5,0,0,6,64
10830 DATA 0,0,0,0,4,0,0
10840 DATA 0,128,0,3,128,0,0
10850 DATA 0,0,0,0,0,0,0
10860 DATA 0,0,0,0,0,0,0
10870 DATA 0,0,0,0,0,0,0
10880 DATA 0,0,0,0,0,0,0
10890 DATA 0,0,0,0,0,0,0
10900 DATA 0,0,0,0,0,0,0
10910 DATA 0,0,0,0,0,0,0
10920 DATA 0,0,1,128,0,6,128
10930 DATA 0,2,64,0,5,192,0
10940 DATA 3,128,0,1,252,0,1
10950 DATA 252,0,1,236,0,1,126
10960 DATA 0,3,248,0,2,120,0
10970 DATA 0,248,0,0,120,0,0
10980 DATA 112,0,0,120,0,0,120
10990 DATA 0,0,48,0,0,48,0
11000 DATA 0,0,0,0,0,0,0
11010 DATA 0,0,0,0,0,0,0
11020 DATA 0,0,0,0,0,0,0
11030 DATA 0,0,0,0,0,0,0
11040 DATA 0,16,0,0,16,0,16
11050 DATA 56,16,10,16,160,4,16
11060 DATA 64,10,124,160,1,255,0
11070 DATA 1,255,0,11,255,144,127
11080 DATA 255,252,11,255,144,1,255
11090 DATA 0,1,255,0,10,124,160
11100 DATA 4,16,64,10,16,160,16
11110 DATA 56,16,0,16,0,0,16
11120 DATA 0,0,0,0,0,0,256
```

Capitolo due

Labirinto rotante

Matt Giwer (la versione per C64 è di Eric Brandon)

«Labirinto rotante» è un tipo differente di labirinto: i suoi muri si muovono in continuazione. Questo gioco impegnativo illustra l'uso degli sprite internamente ad un gioco; comprende anche una spiegazione su come combinare il programma «Shuttle in fuga» con il gioco che presentiamo.

L'obiettivo di «Labirinto rotante» è di portare la nave spaziale dal margine sinistro dello schermo al destro. Iniziate con 2000 unità di carburante che consumate ad un ritmo di 60 unità al secondo sia che lo Shuttle si muova o no. Se toccate un muro o uno dei predoni, perdete 100 unità ogni sessantesimo di secondo. Quando avete esaurito il carburante il gioco ha termine. Fortunatamente potete rifornire i vostri serbatoi raggiungendo l'estremità destra dello schermo.

Se volete interrompere il gioco per un attimo, tenete premuto il tasto SHIFT. Se volete interromperlo più a lungo, usate SHIFT LOCK, ma fate attenzione: si trova proprio accanto al tasto RUN/STOP.

Potete aumentare la velocità di movimento dei muri tenendo premuto il pulsante di sparo del joystick. Questo non farà apparire i varchi più rapidamente, ma aumenterà la velocità di scorrimento dei varchi già presenti. Lo svantaggio consiste nel fatto che mentre il pulsante è premuto, il vostro carburante viene consumato a velocità doppia.

La programmazione di «Labirinto rotante» ha messo in luce dei problemi interessanti. Il primo consiste nei «lampi» — piccole chiazze di «neve» — che appaiono sullo schermo. Normalmente questo non provoca problemi, ma se cercate di usare il registro di controllo delle collisioni fra sprite e sfondo del chip VIC-II, vi accorgerete che gli sprite collidono con i «lampi»!

Il significato di tutto ciò consiste nel fatto che occasionalmente, senza alcuna ragione apparente, lo Shuttle colliderebbe e voi per-

dereste 100 unità di carburante. Dal momento che spostando il set di caratteri si eliminano i lampi, il set suddetto è stato rilocato in $3000.

Un altro cavillo del C64 è rappresentato dal fatto che il chip VIC-II può guardare solo 16Kbyte di memoria contemporaneamente. Quando accendete il vostro calcolatore il chip vede il primo blocco di 16Kbyte dalla locazione $0000 alla locazione $3FFF. È stato deciso, per semplicità, di lasciare le cose in questo stato. Questo significa che i dati degli sprite, il set di caratteri rilocato e l'intero programma BASIC devono essere concentrati in 16Kbyte. A causa di questa limitazione nell'occupazione di memoria, quando il linguaggio macchina crea un set di caratteri alla locazione $3000 distrugge le istruzioni DATA nel programma. Fortunatamente le istruzioni DATA non sono ulteriormente richieste, dal momento che sono già state inserite in memoria tramite istruzioni POKE.

Come si copia il programma

«Labirinto rotante» funzionerà come gioco se il programma 1 verrà copiato correttamente. Dal momento che il fatto di eseguire il programma ne distrugge una parte, siate più che sicuri di aver registrato il programma dopo averlo interamente copiato e prima di eseguirlo.

Se desiderate combinare il gioco «Labirinto rotante» con la presentazione decisamente efficace proposta da «Shuttle in fuga», per ottenere un programma unico, seguite il procedimento sottoindicato.

1. Copiate *l'intero* listato di Shuttle in fuga.
2. Dal programma 1, Labirinto rotante, copiate le righe da 4000 a 4210 e le righe da 11040 a 52010.
3. Copiate il programma 2.
4. Registrate il vostro programma.

Programma 1. Labirinto rotante

```
50 POKE53281,0:POKE53280,0:PRINT"{CLR}"
110 PRINTCHR$(142)
120 IF PEEK(16378)<>16 THEN GOSUB 10000:GO
    SUB 50000
```

```
2000 S=54272
2010 POKES+24,15+16+32:POKES+23,1+16*5
2020 POKES+5,7*16
2030 POKES+6,249
2050 POKES+1,11
4000 V=13*4096
4010 POKE V+21,0
4020 POKES+4,128
4030 FOR I=1 TO 6
4040 POKE V+39+I,7+4*(INT(I/2)<>I/2): POKE
      V+2*I,(36+40*I)AND255:NEXT
4050 POKE V+16,64:POKE 2040,254:POKEV,30:P
     OKEV+1,148
4060 FOR I=2041 TO 2047:POKEI,255:NEXT
4065 POKE V+21,127
4070 PRINT"{CYN}{CLR}G A S
4080 PRINT"02000"
4090 PRINT"PUNTI:"
4100 PRINT"00000"
4110 P(0)=1029:P(4)=1994:P(1)=1039:P(5)=20
     04:P(2)=1049:P(6)=2014:P(3)=1059
4120 SYS 49152
4130 POKE P(0),227
4140 IF PEEK(2)=255 THEN 20000
4150 IF PEEK(653)=1 THEN 4150
4160 IF RND(1)>.05 THEN 4140
4170 IF RND(1)>.5 THEN 4200
4180 P=RND(1)*5:IF PEEK(P(P))<>160 THEN 41
     80
4190 POKE P(P),227:GOTO4140
4200 P=RND(1)*3+4:IF PEEK(P(P))<>160 THEN
     4200
4210 POKE P(P),228:GOTO4140
10000 I=16256:TI$="000000"
10005 PRINT"{HOME}{WHT}{ 12 DES}PRONTO IN"
      LEFT$(STR$(93-INT(TI/60)),4)" SECOND
      I "
10010 READ A:IF A=256 THEN GOTO10025
10020 C1=C1+A:POKE I,A:I=I+1:GOTO 10005
```

```
10025 IF C1<>6062 THEN PRINT"ERRORE NELLA
      {SPAZI}SOMMA DI CONTROLLO DI RIGA 10
      025":END
10026 RETURN
10030 DATA 0,0,0,0,0,0,0
10040 DATA 0,0,24,0,0,28,0
10050 DATA 0,31,0,0,31,255,240
10060 DATA 31,255,8,20,255,254,31
10070 DATA 127,255,30,63,254,24,0
10080 DATA 0,0,0,0,0,0,0
10090 DATA 0,0,0,0,0,0,0
10100 DATA 0,0,0,0,0,0,0
10110 DATA 0,0,0,0,0,0,0,0
11030 DATA 0,0,0
11040 DATA 0,16,0,0,16,0,16
11050 DATA 56,16,10,16,160,4,16
11060 DATA 64,10,124,160,1,255,0
11070 DATA 1,255,0,11,255,144,127
11080 DATA 255,252,11,255,144,1,255
11090 DATA 0,1,255,0,10,124,160
11100 DATA 4,16,64,10,16,160,16
11110 DATA 56,16,0,16,0,0,16
11120 DATA 0,0,0,0,0,0,256
20000 SC=0:FOR I=0 TO 4:SC=SC+(PEEK(1148-I
      )-48)*10↑I:NEXT I
20010 IF H<SC THEN H=SC
20020 POKE S+4,128
20030 POKE 13*4096+21,0
20040 FOR I=1 TO 1000:NEXT I
20050 PRINT"{CLR}CARBURANTE ESAURITO...
      {GIU'}
20060 PRINT"HAI TOTALIZZATO{WHT}"SC"{CYN}P
      UNTI
20070 PRINT"RECORD{WHT}"H"{CYN}
20080 PRINT"{ 3 GIU'}{ 10 SPAZI}ANCORA? (S
       O N)"
20090 PRINT"{GIU'} OPPURE PREMI IL PULSANT
      E DI SPARO PER{ 3 SPAZI}RIPARTIRE"
20100 GETA$
```

```
20110 IF A$="N"THEN END
20120 IF (PEEK(56320) AND 16)=0 THEN GOTO4
      000
20130 IF A$<>"S" THEN 20100
20140 GOTO4000
50000 I=49152:TI$="000000"
50010 PRINT"{HOME}{WHT}{ 12 DES}PRONTO IN"
      LEFT$(STR$(86-INT(TI/60)),4)" SECOND
      I "
50015 READ A:IF A=256 THEN PRINT"{HOME}
      { 10 DES}{ 21 SPAZI}{SPAZI}":GOTO500
      45
50020 IF A=-1 THEN I=49920 : GOTO 50010
50030 IF A=-2 THEN I=50688 : GOTO 50010
50040 C2=C2+A:POKE I,A:I=I+1:GOTO 50010
50045 IF C2<>188431 THEN PRINT"ERRORE NELL
      A SOMMA DI CONTROLLO IN 50045":END
50046 RETURN
50050 DATA 120,169,0,141,20,3,169
50060 DATA 195,141,21,3,88,173,14
50070 DATA 220,41,254,141,14,220,165
50080 DATA 1,41,251,133,1,160,0
50090 DATA 185,0,208,153,0,48,185
50100 DATA 0,50,153,0,50,185,0
50110 DATA 209,153,0,49,185,0,211
50120 DATA 153,0,51,185,0,212,153
50130 DATA 0,52,185,0,213,153,0
50140 DATA 53,185,0,214,153,0,54
50150 DATA 185,0,215,153,0,55,169
50160 DATA 15,141,156,200,200,208,200
50170 DATA 165,1,9,4,133,1,173
50180 DATA 14,220,9,1,141,14,220
50190 DATA 169,28,141,24,208,169,15
50200 DATA 141,156,200,169,255,141,15
50210 DATA 212,169,128,141,18,212,169
50220 DATA 0,133,2,141,224,207,141
50230 DATA 255,207,141,254,207,141,253
50240 DATA 207,141,252,207,141,249,207
50250 DATA 160,6,169,20,153,0,207
```

```
50260 DATA 169,0,153,16,207,136,208
50270 DATA 243,169,251,141,251,207,160
50280 DATA 0,169,4,133,252,132,251
50290 DATA 169,216,133,254,132,253,169
50300 DATA 160,160,5,145,251,160,10
50310 DATA 145,251,160,15,145,251,160
50320 DATA 20,145,251,160,25,145,251
50330 DATA 160,30,145,251,160,35,145
50340 DATA 251,165,251,24,105,40,133
50350 DATA 251,144,2,230,252,201,232
50360 DATA 208,211,169,1,160,10,145
50370 DATA 253,169,4,160,5,145,253
50380 DATA 169,7,160,15,145,253,169
50390 DATA 14,160,20,145,253,169,8
50400 DATA 160,25,145,253,169,13,160
50410 DATA 30,145,253,169,3,160,35
50420 DATA 145,253,165,253,24,105,40
50430 DATA 133,253,144,2,230,254,201
50440 DATA 232,208,199,96,-1
50450 DATA 173,141
50460 DATA 2,201,1,208,3,76,49
50470 DATA 234,230,2,165,2,201,2
50480 DATA 240,3,76,49,234,169,0
50490 DATA 133,2,169,3,133,252,169
50500 DATA 216,133,251,160,45,177,251
50510 DATA 32,79,195,160,55,177,251
50520 DATA 32,79,195,160,65,177,251
50530 DATA 32,79,195,160,75,177,251
50540 DATA 32,79,195,165,251,24,105
50550 DATA 40,133,251,144,2,230,252
50560 DATA 201,192,208,213,76,0,198
50570 DATA 201,160,240,19,201,32,240
50580 DATA 37,162,1,232,221,174,195
50590 DATA 208,250,202,189,174,195,145
50600 DATA 251,96,152,56,233,40,168
50610 DATA 177,251,201,32,240,1,96
50620 DATA 152,24,105,40,168,169,227
50630 DATA 145,251,96,165,252,201,3
50640 DATA 240,22,152,56,233,40,168
```

```
50650 DATA 177,251,201,160,240,1,96
50660 DATA 152,24,105,40,168,169,99
50670 DATA 145,251,96,152,24,105,120
50680 DATA 168,177,251,201,100,240,1
50690 DATA 96,152,56,233,120,168,169
50700 DATA 99,145,251,96,160,228,239
50710 DATA 249,226,120,119,99,32,32
50720 DATA 100,111,121,98,248,247,227
50730 DATA -2,169,7,133,252
50740 DATA 169,32,133,251,160,170,177
50750 DATA 251,32,47,198,160,180,177
50760 DATA 251,32,47,198,160,190,177
50770 DATA 251,32,47,198,165,251,56
50780 DATA 233,40,133,251,176,2,198
50790 DATA 252,201,56,208,220,76,160
50800 DATA 198,201,160,240,19,201,32
50810 DATA 240,37,162,1,232,221,142
50820 DATA 198,208,250,202,189,142,198
50830 DATA 145,251,96,152,24,105,40
50840 DATA 168,177,251,201,32,240,1
50850 DATA 96,152,56,233,40,168,169
50860 DATA 228,145,251,96,165,251,201
50870 DATA 32,240,22,152,24,105,40
50880 DATA 168,177,251,201,160,240,1
50890 DATA 96,152,56,233,40,168,169
50900 DATA 100,145,251,96,152,56,233
50910 DATA 120,168,177,251,201,99,240
50920 DATA 1,96,152,24,105,120,168
50930 DATA 169,100,145,251,96,32,99
50940 DATA 119,120,226,249,239,228,160
50950 DATA 160,227,247,248,98,121,111
50960 DATA 100,32,173,0,220,72,41
50970 DATA 15,201,15,240,8,169,129
50980 DATA 141,4,212,76,183,198,169
50990 DATA 128,141,4,212,104,41,16
51000 DATA 205,255,207,240,48,141,255
51010 DATA 207,201,16,208,24,169,2
51020 DATA 141,15,195,169,1,141,252
51030 DATA 198,141,229,200,169,0,141
```

```
51040 DATA 250,207,141,224,207,76,239
51050 DATA 198,169,1,141,15,195,169
51060 DATA 2,141,252,198,141,229,200
51070 DATA 169,0,133,2,32,245,198
51080 DATA 76,32,200,238,250,207,173
51090 DATA 250,207,201,1,240,1,96
51100 DATA 169,0,141,250,207,173,0
51110 DATA 220,141,254,207,41,1,208
51120 DATA 13,173,253,207,201,253,240
51130 DATA 23,206,253,207,76,45,199
51140 DATA 173,254,207,41,2,208,10
51150 DATA 173,253,207,201,3,240,3
51160 DATA 238,253,207,173,254,207,41
51170 DATA 8,208,13,173,252,207,201
51180 DATA 3,240,23,238,252,207,76
51190 DATA 82,199,173,254,207,41,4
51200 DATA 208,10,173,252,207,201,253
51210 DATA 240,3,206,252,207,173,254
51220 DATA 207,41,3,201,3,208,16
51230 DATA 173,253,207,240,11,16,6
51240 DATA 238,253,207,76,107,199,206
51250 DATA 253,207,173,254,207,41,12
51260 DATA 201,12,208,16,173,252,207
51270 DATA 240,11,16,6,238,252,207
51280 DATA 76,132,199,206,252,207,174
51290 DATA 249,207,208,32,174,240,207
51300 DATA 224,60,176,25,173,253,207
51310 DATA 24,109,1,208,201,80,176
51320 DATA 5,169,244,76,191,199,201
51330 DATA 244,144,27,169,80,76,191
51340 DATA 199,173,253,207,24,109,1
51350 DATA 208,201,41,176,5,169,244
51360 DATA 76,191,199,201,244,144,2
51370 DATA 169,41,141,1,208,173,252
51380 DATA 207,48,32,24,109,0,208
51390 DATA 141,240,207,173,249,207,105
51400 DATA 0,141,249,207,201,1,208
51410 DATA 42,173,240,207,201,55,144
51420 DATA 35,32,155,200,76,4,200
```

```
51430 DATA 24,109,0,208,141,240,207
51440 DATA 173,249,207,105,255,141,249
51450 DATA 207,208,12,173,240,207,201
51460 DATA 25,176,5,169,25,141,240
51470 DATA 207,173,240,207,141,0,208
51480 DATA 173,16,208,41,254,13,249
51490 DATA 207,141,16,208,173,31,208
51500 DATA 41,1,240,3,76,101,200
51510 DATA 96,162,5,189,119,4,201
51520 DATA 57,240,6,254,119,4,76
51530 DATA 58,200,169,48,157,119,4
51540 DATA 202,208,235,76,58,200,162
51550 DATA 5,189,39,4,201,48,240
51560 DATA 6,222,39,4,76,222,200
51570 DATA 169,57,157,39,4,202,208
51580 DATA 235,120,169,234,141,21,3
51590 DATA 169,49,141,20,3,88,169
51600 DATA 255,133,2,76,222,200,0
51610 DATA 162,0,160,240,238,32,208
51620 DATA 232,208,250,200,208,247,169
51630 DATA 0,141,32,208,162,3,189
51640 DATA 39,4,201,48,240,4,222
51650 DATA 39,4,96,169,57,157,39
51660 DATA 4,202,208,237,162,5,169
51670 DATA 48,157,39,4,202,208,250
51680 DATA 104,104,76,81,200,160,15
51690 DATA 162,3,189,39,4,201,57
51700 DATA 240,6,254,39,4,76,180
51710 DATA 200,169,48,157,39,4,202
51720 DATA 208,235,136,208,230,169,0
51730 DATA 141,249,207,169,25,141,240
51740 DATA 207,169,148,141,1,208,172
51750 DATA 156,200,192,9,240,4,136
51760 DATA 140,156,200,173,5,4,201
51770 DATA 160,208,5,169,227,141,5
51780 DATA 4,96,238,224,207,173,224
51790 DATA 207,201,1,240,3,76,124
51800 DATA 201,169,0,141,224,207,173
51810 DATA 27,212,201,7,176,25,168
```

```
51820 DATA 185,0,207,201,20,208,8
51830 DATA 169,1,153,16,207,76,16
51840 DATA 201,201,255,208,5,169,255
51850 DATA 153,16,207,160,6,185,0
51860 DATA 207,24,121,16,207,153,0
51870 DATA 207,72,152,10,170,104,157
51880 DATA 1,208,136,208,235,160,6
51890 DATA 185,0,207,201,20,240,10
51900 DATA 201,255,240,6,136,208,242
51910 DATA 76,66,201,169,0,153,16
51920 DATA 207,76,52,201,173,30,208
51930 DATA 41,1,240,51,162,0,160
51940 DATA 240,238,32,208,232,208,250
51950 DATA 200,208,247,169,0,141,32
51960 DATA 208,162,3,189,39,4,201
51970 DATA 48,240,6,222,39,4,76
51980 DATA 49,234,169,57,157,39,4
51990 DATA 202,208,235,162,5,169,48
52000 DATA 157,39,4,202,208,250,76
52010 DATA 49,234,256
```

Programma 2. Come collegare Shuttle in fuga a labirinto rotante

```
90 POKE45,15000AND255:POKE46,15000/256:CLR

91 REM NIENTE SPAZI IN RIGA 90 !! MOLTO IM
   PORTANTE !
120 IFPEEK(49153)<>169THENGOSUB10000:GOSUB
    500000:C2=0
720 GOTO4000
4005 POKEV+23,0
4020 POKES+5,7*16:POKES+6,249:POKES+4,128
4050 POKEV+16,64:POKE2040,244:POKEV,30:POK
     EV+1,148
10005 PRINT"{HOME}{WHT}{ 12 DES}PRONTO IN"
      LEFT$(STR$(146-INT(TI/60)),4)" SECON
      DI "
```

```
10010 READA:IFA=256THEN10025
10025 IFC1<>34430THENPRINT"ERRORE NELLA SO
      MMA DI CONTROLLO IN RIGA 10025":END
11030 DATA0,0,0,0,0,0,0
50010 PRINT"{HOME}{WHT}{ 12 DES}PRONTO IN"
      LEFT$(STR$(101-INT(TI/60)),4)" SECON
      DI "
```

Effetti sonori

Capitolo tre

Come ravvivare i programmi con gli effetti sonori

Gregg Peele

Siete stati in una sala giochi di recente? Se lo avete fatto, allora conoscete l'impatto che gli effetti sonori hanno sul realismo di un video gioco. Sibili, spari ed esplosioni di tutti i tipi sono mescolati con motivi ed altri effetti speciali. Benché siano gli effetti visivi a fornire la maggior parte degli stimoli in un gioco, buoni effetti sonori aggiungono quel certo tocco professionale.

Come si possono effettivamente utilizzare gli effetti sonori internamente ad un programma? Naturalmente, collisioni, esplosioni ed altri eventi emozionanti che avvengono sullo schermo necessitano dell'aggiunta del realismo dato dagli effetti sonori. Ma non limitate il loro uso a questi effetti speciali.

Gli effetti sonori possono aggiungere uno sprazzo di interesse a sezioni del programma particolarmente noiose. Può darsi che siano necessari 10 o 20 secondi per predisporre lo schermo per il vostro gioco. Aggiungendo gli effetti sonori a questa parte del programma, potete mantenere desto l'interesse anche se, sotto l'aspetto puramente visivo, non sta succedendo alcunché di interessante.

Gli effetti sonori possono anche servire ad altri scopi di carattere più pratico all'interno di certi tipi di programma. Un breve cicalino può segnalare una condizione di errore o ricordare all'utente di prestare attenzione al calcolatore.

Fortunatamente la Commodore ha inserito ottime capacità sonore nel C64. In effetti, esso contiene uno dei più sofisticati sistemi di produzione del suono di tutti i personal computer, un vero e proprio sintetizzatore su di un solo chip.

Fanfara

Di seguito è riportato un programma il quale crea un effetto

sonoro che può essere usato per aggiungere un po' di eccitazione a quasi tutti i programmi. Il sottoprogramma riproduce il suono di una fanfara sul tipo di quelle delle sale giochi, adatta per qualche trionfale momento di un gioco.

L'aggiunta del suono può migliorare praticamente qualsiasi programma per calcolatore. Non dimenticate l'ulteriore dimensione che gli effetti sonori possono aggiungere alle vostre capacità di programmazione.

Fanfara

```
10 REM FANFARA
20 B=54272:FORCLEAR=B TO B+24:POKE CLEAR,0
   :NEXT
30 FOR R=1 TO 4
40 POKE B+5,85:POKE B+6,85:POKE B+12,85:PO
   KE B+13,85
50 POKE B+24,15:POKE B+4,33:POKE B+11,17
60 FOR X=1 TO 6:READ H1,L1,H2,L2:POKE B+1,
   H1:POKE B,L1
70 POKE B+8,H2:POKE B+7,L2
80 IF H1=50 THEN FOR T=1 TO 200:NEXT
90 FOR T=1 TO 100:NEXT
100 DATA 25,30,18,209,33,135,25,30,42,62,3
    1,165,50,60,37,162,42,62,31,165,50,60
110 DATA 37,162
120 NEXT X
130 POKE B+4,32:POKE B+11,16:FOR W=1 TO 50
    0:NEXT
140 RESTORE:NEXT R
150 FORCLEAR=B TO B+24:POKE CLEAR,0:NEXT
```

Impariamo a generare effetti sonori, Parte 1.

Gregg Peele

Questo articolo vi aiuterà ad imparare a creare effetti sonori con il C64. Inoltre contiene un programma di utilità pratica, che facilita la programmazione sonora sul C64 e l'aggiunta degli effetti sonori ai vostri programmi.

Il meraviglioso chip SID

Il C64 ha tre *voci* indipendenti (canali sonori), ciascuna avente una di quattro possibili *forme d'onda* (toni). Queste voci, prodotte dal chip SID MOS 6581 (Sound Interface Device, dispositivo d'interfaccia sonora), possono essere predisposte per simulare quasi ogni suono. In effetti, le capacità del chip SID sono state paragonate a quelle di sintetizzatori aventi un prezzo superiore a quello dell'intero C64. Per capire come si usa in pratica il chip SID è necessaria una breve trattazione della natura degli effetti sonori.

Un po' di teoria del suono

La maggior parte dei suoni in campo musicale ed in natura hanno un *tono* ben definito. Il tono non è che un modo di descrivere quanto un suono sia alto o basso.

Il chip SID ha un'ampiezza tonale di nove ottave. Vale a dire circa due ottave più di un pianoforte. Quando si scrivono programmi questi valori di tono sono formati da due *byte* (un byte è una locazione di memoria che può contenere un valore da 0 a 255). Questo fornisce un campo di valori con più di 65000 (256x256) possibilità di ottenere valori differenti di tono per le note. La forma d'onda *pulse (vibrato),* una delle quattro disponibili, consente per-

sino un più vasto campo di valori del tono.

Forme d'onda

Dal momento che abbiamo già citato un paio di volte le forme d'onda, dovremmo chiarire esattamente che cosa sia una forma d'onda. Quasi ogni suono consiste di un movimento pulsante cui generalmente si fa riferimento come vibrazione. Materiali diversi vibrano in modo diverso. Questa è una delle ragioni per cui i diversi strumenti dell'orchestra hanno qualità timbriche particolari. Il chip SID è in grado di produrre quattro diverse forme d'onda: triangolare, dente di sega, vibrato e rumore. Ciascuna di queste forme d'onda produce un suono unico e, assieme al controllo del tono e dell'involuzione, forma il meccanismo di base della sintesi sonora con il Commodore 64.

Un sasso nell'acqua

Le onde sonore, come le onde provocate da un sasso gettato in uno stagno, cambiano costantemente. In effetti, gran parte della nostra capacità nel distinguere un suono dall'altro è data dalla caratteristica esclusiva della variazione che identifica ciascun suono. Un esempio familiare è costituito dai suoni differenti che vengono provocati quando si colpisce qualcosa con un martello di metallo o di gomma. La maggior parte del suono prodotto dal martello di gomma viene assorbita all'interno del martello stesso.

Involuzioni

La maggior parte dei suoni segue una forma simile con l'andare del tempo. Questa forma è *l'involuzione.* In un primo momento l'evento iniziale che crea il suono manda il livello del volume rapidamente verso l'alto. Questa sezione dell'involuzione, chiamata *l'attaccare,* può essere il fattore principale di definizione del suono. Un battito di mani consiste quasi interamente della sezione attaccare.

Dopo questo attacco iniziale il livello del volume decresce durante la sezione *decadere.* Dopo questo decremento il livello del volume si stabilizza per qualche tempo in quella che viene chiamata la sezione *sostenere.* Il suono inizia quindi la sua discesa finale che termina nel silenzio. Questa discesa è la parte del suono chia-

mata *rilasciare.*

La combinazione di attaccare, decadere, sostenere e rilasciare è l'involuzione, chiamata a volte involuzione ADSR. Il chip SID fornisce un sistema di definizione del modo in cui il suono evolve nel tempo. Questa evoluzione è controllata da un *generatore di involuzione.* Le sezioni attaccare e decadere sono controllate internamente ad un byte; ciascuna fa uso di quattro bit (in ogni byte ci sono otto cifre binarie, o bit). Il valore contenuto in questo byte stabilisce il tasso di variazione del volume attraverso il tempo. Un valore numerico basso per attaccare e decadere indica una breve durata per questa particolare sezione. Un valore più alto incrementa la durata di una particolare sezione.

Anche le parti sostenere e rilasciare dell'involuzione si dividono un byte. Tuttavia, sostenere non fa riferimento ad un valore temporale, ma ad un livello di volume. La sezione rilasciare, come attaccare e decadere, fa riferimento ad un tasso di variazione ed i valori per questa sezione cambiano la quantità di tempo destinata alla variazione suddetta.

Per ammissione generale tutto ciò non è facilmente comprensibile in un primo tempo. Se copiate e mandate in esecuzione il programma 1, vedrete ed ascolterete una dimostrazione animata dell'involuzione ADSR.

Tutti insieme, ora!

La produzione di effetti sonori con il chip SID richiede che certi registri (locazioni di memoria) all'interno del chip contengano dei valori numerici, che rappresentino la forma d'onda, il volume e l'involuzione ADSR. Inoltre devono esserci dei dispositivi che consentano di fissare la lunghezza delle note. In BASIC vengono utilizzati dei comandi POKE per inserire valori numerici per le forme d'onda, il volume e l'ADSR nelle locazioni opportune.

La lunghezza dell'effetto sonoro è determinata dall'uso di due cicli BASIC FOR/NEXT come temporizzatori. Quanto più il limite superiore del ciclo è alto, tanto più lunga è la durata di quella particolare parte dell'effetto sonoro. Il primo ciclo fissa la quantità di tempo concessa per la parte sostenere dell'effetto sonoro ed il secondo ciclo la quantità concessa per la parte decadere. Il byte della forma d'onda attiva l'effetto sonoro. Quando viene disattivato inizia a decadere, il che pone termine all'effetto sonoro. Un bit del byte, cui si fa riferimento come bit di *uscita,* è riservato a questo

scopo.

Ecco la serie degli eventi: innanzi tutto i valori numerici per il volume e l'ADSR vengono posti nelle locazioni opportune usando dei comandi POKE. Quindi date il via all'effetto sonoro, attivando il byte della forma d'onda ponendo a 1 il bit di uscita (questo byte conterrà sempre un valore numerico dispari, dal momento che il bit d'uscita è il meno significativo all'interno del byte). Ora viene utilizzato il nostro ciclo FOR/NEXT per provocare un ritardo, che si esaurisce mentre vengono eseguite le sezioni attaccare, decadere e sostenere. Quando questo ciclo termina rimpiazziamo il valore numerico contenuto nel byte della forma d'onda con un valore equivalente meno uno. Ciò ripristina il bit d'uscita e segnala l'inizio della sezione rilasciare. Il volume decresce finché l'effetto sonoro si smorza, infine, nel silenzio. Un altro ciclo FOR/NEXT consente alla sezione rilasciare un tempo di esecuzione adeguato.

Un programma esemplificativo

Vi sembra che tutto ciò sia disperatamente complicato? Per chiarire meglio le forme d'onda, i toni ed il generatore di involuzioni ho accluso un programma che vi permette di manipolare tutti i parametri citati e di creare effettivamente i vostri sottoprogrammi di effetti sonori, per poterli utilizzare in altri programmi. Per usare il programma 2 limitatevi ad inserire i valori numerici per il volume, la forma d'onda, l'ADSR (attaccare, decadere, sostenere, rilasciare) e i valori numerici della lunghezza di sostenere e rilasciare (ricordatevi che, all'interno del campo di valori dati, i valori numerici più bassi rappresentano o volumi di livello inferiore o intervalli di tempo di durata inferiore per ciascuna sezione).

Dovete anche inserire due valori numerici per definire il tono del timbro. Questo valore di tono può essere ricavato dalla tavola di valori visualizzata sullo schermo o da quella riportata nella «Guida di riferimento del Programmatore» del C64.

Quando vi si pone la domanda ANCORA? premete N, se vi piace l'effetto sonoro che avete prodotto o S, se volete continuare a modificare l'effetto sonoro. Se premete N, verrà creato un sottoprogramma che potete aggiungere ai vostri programmi. Vi si chiederà il numero che contraddistingue la riga iniziale e lo scarto che volete lasciare tra le righe del sottoprogramma. Quindi il vostro sottoprogramma sonoro apparirà sullo schermo (prima di battere N assicuratevi di aver registrato il programma originale, dal momento

che verrà cancellato). Potete ora usare questo sottoprogramma sonoro in qualsiasi programma o registrarlo su disco o nastro per uso futuro.

Un piccolo passo

Abbiamo compiuto solo il primo passo verso la comprensione delle complessità e delle possibilità del chip SID. Il programma utilizza solo una delle tre voci del C64 e dobbiamo ancora trattare alcune applicazioni avanzate delle caratteristiche del chip SID. Tuttavia abbiamo compiuto un grande passo avanti nel nostro tentativo di scoprire i meccanismi della sintesi sonora sul Commodore 64.

Programma 1. Involuzione ADSR

```
5 PRINT"{CLR}":POKE53281,12:POKE646,0
10 PRINTTAB(8)CHR$(18)CHR$(169)CHR$(223)"
   {OFF} "
20 PRINTTAB(7)CHR$(18)CHR$(169)"
   { 2 SPAZI}"CHR$(223)
30 PRINTTAB(6)CHR$(18)CHR$(169)"
   { 4 SPAZI}"CHR$(223)
40 PRINTTAB(5)CHR$(18)CHR$(169)"
   { 6 SPAZI}"CHR$(223)
50 PRINTTAB(4)CHR$(18)CHR$(169)"
   { 19 SPAZI}"CHR$(223)
60 PRINTTAB(3)CHR$(18)CHR$(169)"
   { 21 SPAZI}"CHR$(223)
70 PRINTTAB(2)CHR$(18)CHR$(169)"
   { 23 SPAZI}"CHR$(223)
80 PRINTTAB(1)CHR$(18)CHR$(169)"
   { 25 SPAZI}"CHR$(223)
90 PRINT
100 PRINT"{ 4 SPAZI}A{ 4 SPAZI}D
    { 3 SPAZI}SOSTENERE{ 3 SPAZI}R
105 PRINT"{ 4 SPAZI}T{ 4 SPAZI}E
    { 15 SPAZI}I
```

```
110 PRINT"{ 4 SPAZI}T{ 4 SPAZI}C
    { 15 SPAZI}L
115 PRINT"{ 4 SPAZI}A{ 4 SPAZI}A
    { 15 SPAZI}A
120 PRINT"{ 4 SPAZI}C{ 4 SPAZI}D
    { 15 SPAZI}S
125 PRINT"{ 4 SPAZI}C{ 4 SPAZI}E
    { 15 SPAZI}C
130 PRINT"{ 4 SPAZI}A{ 4 SPAZI}R
    { 15 SPAZI}I
150 PRINT"{ 4 SPAZI}R{ 4 SPAZI}E
    { 15 SPAZI}A
160 PRINT"{ 4 SPAZI}E{ 20 SPAZI}R
165 PRINT"{ 25 SPAZI}E
170 CL=55296:S=54272:W=S+4:AD=S+5:SR=S+6:V
    =S+24
175 POKEV,15:POKEAD,202:POKESR,58:POKES,13
    5:POKES+1,33:POKEW,33
180 FORR=CLTOCL+5:FORU=RTOCL+1024STEP40
185 POKEU,1:NEXT:NEXT
190 FORR=CL+6TOCL+12:FORU=RTOCL+1024STEP40

195 POKEU,1:NEXT:NEXT
197 FORR=CL+13TOCL+23:FORU=RTOCL+1024STEP4
    0
199 POKEU,1:NEXT:NEXT
200 POKEW,16:FORR=CLTOCL+28:FORU=RTOCL+102
    4STEP40
290 POKEU,1:NEXT:NEXT
300 FORT=STOS+28:POKET,0:NEXT
```

Programma 2. Compositore 1

```
5 POKE53281,1:POKE646,0
10 S=54272:FORE=STOS+28:POKEE,0:NEXT
15 PRINT"{CLR}{SU}":GOSUB200
20 INPUT"TASSO DI ATTACCARE 0-15";AT
23 INPUT"TASSO DI DECADERE 0-15";DE:AD=16*
   AT+DE:POKE54277,AD
```

```
25 INPUT"VOLUME DI SOSTENERE 1-15";SU
27 INPUT"TASSO DI RILASCIARE 0-15";RL:J=16
   *SU+RL
30 POKE54278,J:INPUT"VOLUME GENERALE 1-15"
   ;V:POKE54296,V
32 INPUT"BYTE PIU' SIGNIFICATIVO";H
33 INPUT"BYTE MENO SIGNIFICATIVO";L:POKE54
   273,H :POKE54272,L
34 INPUT"DURATA DI SOSTENERE (* .1 SECOND)
   ";LE:LE=LE*100
35 INPUT"FORMA D'ONDA 17,33,65,OR 129 ";W:
   IFW=65THENGOSUB350:GOTO39
38 PRINT"{GIU'}"
39 POKE54276,W
40 FORT=1TOLE:NEXTT
42 POKE54276,(W-1)
43 FORT=1TODL:NEXT
50 S=54272
60 PRINT"{CLR}{ 12 GIU'}{RVS}ANCORA O CANC
   ELLARE ?{OFF} S O N"
65 GETA$:IFA$="C"THENFORAS=54272TO54272+24
   :POKEAS,0:NEXT
70 IFA$="S"THENPRINT"{HOME}{ 12 GIU'}
   { 24 SPAZI}":GOTO20
75 IFA$<>"N"THEN65
80 REM PRINT PROGRAM
85 INPUT"{CLR}RIGA INIZIALE";SL:INPUT"INCR
   EMENTO";IN
86 PRINT"{CLR}"
88 PRINT"{ 3 GIU'}NEW{ 3 GIU'}"
89 PRINTSL;"S=54272:FORE=STOS+28:POKEE,0:N
   EXT":SL=SL+IN
90 PRINTSL;"POKE54296,";V;":POKE54277,";AD
   ;":POKE54278,";J:SL=SL+IN
100 IFW=65THENPRINTSL;"POKE54275,";PW;":PO
    KE54274,";P2:SL=SL+IN
120 PRINTSL;"POKE 54273,";H;":POKE54272,";
    L;":POKE54276,";W:SL=SL+IN
140 PRINTSL;"FORT=1TO";LE;":NEXT";":POKE54
```

```
    276,";(W-1)
155 PRINT"{HOME}";:FORR=631TO644:POKER,13:
    NEXT
160 POKE198,13
165 END
200 PRINT"DATI ESEMPLIFICATIVI PER VALORI
    DEL TONO"
205 PRINT" TONO{ 2 SPAZI}BYTE +{ 2 SPAZI}
    BYTE -{ 2 SPAZI}{RVS}FORMA D'ONDA
210 PRINT"{ 2 SPAZI}DO{ 5 SPAZI}33
    { 5 SPAZI}135{ 4 SPAZI}TRIANGOLARE=17
220 PRINT"{ 2 SPAZI}DO#{ 4 SPAZI}35
    { 5 SPAZI}134{ 4 SPAZI}DENTE DI SEGA=3
    3
230 PRINT"{ 2 SPAZI}RE{ 5 SPAZI}37
    { 5 SPAZI}162{ 4 SPAZI}RUMORE=129
240 PRINT"{ 2 SPAZI}RE#{ 4 SPAZI}39
    { 5 SPAZI}223{ 4 SPAZI}VIBRATO=65
250 PRINT"{ 2 SPAZI}MI{ 5 SPAZI}42
    { 6 SPAZI}62
260 PRINT"{ 2 SPAZI}FA{ 5 SPAZI}44
    { 5 SPAZI}193
270 PRINT"{ 2 SPAZI}FA#{ 4 SPAZI}47
    { 5 SPAZI}107
280 PRINT"{ 2 SPAZI}SOL{ 4 SPAZI}50
    { 6 SPAZI}60
290 PRINT"{ 2 SPAZI}SOL#{ 3 SPAZI}53
    { 6 SPAZI}57
300 PRINT"{ 2 SPAZI}LA{ 5 SPAZI}56
    { 6 SPAZI}99
335 PRINT
340 RETURN
350 INPUT"AMPIEZZA DELLA VIBRAZIONE SUPERI
    ORE{ 5 SPAZI}(1-15)";PW
360 INPUT"AMPIEZZA DELLA VIBRAZIONE INFERI
    ORE{ 5 SPAZI}(0-255)";P2
370 POKE54275,PW:POKE54274,P2:RETURN
```

Impariamo a generare effetti sonori, Parte 2.

Gregg Peele

Avete mai desiderato di creare proprio l'effetto sonoro adatto per un gioco? Od il tono giusto per una canzone? La conclusione di questo articolo in due parti ed il programma di utilità pratica che lo accompagna potrebbero essere proprio ciò di cui avete bisogno per creare nuovi interessanti effetti sonori col vostro C64.

Nella prima parte abbiamo esplorato alcune delle cognizioni fondamentali per la produzione degli effetti sonori sul Commodore 64. Abbiamo preso in esame l'ADSR (attaccare, decadere, sostenere e rilasciare) ed utilizzato questi parametri insieme al volume, al tono ed alla forma d'onda per produrre effetti sonori differenti. In questa seconda parte esamineremo ancora più a fondo le capacità del sintetizzatore incorporato su un solo chip del C64, il dispositivo di interfaccia sonora (Sound Interface Device, SID). Parleremo di filtri, modulazione sonora e sincronizzazione e presenteremo un programma di utilità pratica, «Compositore 2», che faciliterà l'uso di queste tecniche all'interno dei vostri programmi.

Avete cambiato i filtri di recente?

Il chip SID del C64 ha tre filtri, ma, contrariamente ai filtri della vostra automobile, non dovrebbe essere mai necessario sostituirli. Tuttavia essi condividono alcune caratteristiche dei filtri delle automobili. Proprio come un filtro dell'olio permette a questo di passare, intercettando altre particelle indesiderate, i filtri del chip SID lasciano passare parte degli effetti sonori, *filtrando* in maniera selettiva la parte restante dell'effetto sonoro. I filtri dei sintetizzatori forniscono un mezzo di manipolazione dei suoni per produrre effetti differenti.

Questi tre filtri sono chiamati *passa-alto, passa-basso* e *passa-banda* (vedi le Figure da 1 a 3). Il filtro passa-alto è progettato per escludere le frequenze più basse, lasciando passare le più alte. Il filtro passa-basso ha l'effetto opposto: esclude le alte frequenze, lasciando passare le basse. Il filtro passa-banda consente il passaggio di una banda o gruppo di frequenze, mentre le frequenze al di sopra ed al di sotto della banda vengono soppresse.

Il filtro che avete scelto viene attivato ponendo a 1 i bit 4 (passa-basso), 5 (passa-banda) o 6 (passa-alto) nel registro 24 del SID (Vedi «Impariamo la grafica «bitmap» — al Capitolo 2 del Volume 1° — per maggiori dettagli sull'attivazione ed il disinserimento dei bit). Questi filtri possono essere usati in combinazione per effetti aggiuntivi. Ad esempio, inserire i filtri passa-basso e passa-alto insieme crea l'effetto opposto del filtro passa-banda; solo le frequenze più alte e più basse passano inalterate, mentre le frequenze mediane vengono soppresse.

L'entità dell'effetto sonoro che viene escluso da un filtro è fissata dalla *frequenza di taglio.* Il filtro taglia l'effetto sonoro a partire da questa frequenza. La frequenza di taglio dei filtri è controllata dai tre bit meno significativi del registro 21 del SID e da tutti gli otto bit del registro 22. Alcuni degli effetti sonori più interessanti che è possibile ottenere col C64 vengono creati incrementando o decrementando questa serie di bit mentre viene prodotto un effetto sonoro. Volete ottenere il suono di una nave spaziale aliena che atterra? Usate l'effetto sonoro normale della vostra nave spaziale, aggiungete un filtro ed aumentate o diminuite gradualmente questi otto bit durante la discesa della vostra astronave. Una certa combinazione di forme d'onda ed il cambiamento dei filtri possono creare proprio l'effetto adatto per una nave spaziale che atterra.

Sintesi additiva e sottrattiva

L'uso dei filtri è un esempio di *sintesi sottrattiva.* La sintesi sottrattiva rappresenta un metodo di manipolazione dei suoni tramite sottrazione di parti di un singolo effetto sonoro, mettendo in evidenza altre parti che normalmente può darsi non vengano percepite. La *sintesi additiva,* invece, congiunge due suoni per ottenere un suono unico nuovo. Sia la *modulazione sonora* che la *sincronizzazione* costituiscono esempi di sintesi additiva.

La modulazione sonora

La modulazione sonora rappresenta una forma di sintesi sonora

additiva, che cambia in maniera impressionante le qualità timbriche e tonali di due toni. Motivi che sono stati fatti passare attraverso la modulazione sonora non mantengono il loro tono e timbro originale. Invece la somma e la differenza delle due frequenze vengono conservate. Ad esempio, se il primo effetto sonoro è un tono che vibra a 100 vibrazioni al secondo (v/s) ed il secondo vibra a 200 v/s, il tono di modulazione sonora sarà una combinazione della somma (300 v/s) e della differenza (100 v/s).

Normalmente i toni ottenuti dalla modulazione sonora forniscono effetti sonori molto differenti da quelli dei due toni originali. Dal momento che la maggior parte dei toni musicali è costituita da complessi fenomeni, composti da parecchie frequenze minori meno ovvie (armoniche), il tono ottenuto dalla modulazione sonora può essere realmente molto complesso dal punto di vista timbrico.

Per ottenere la modulazione sonora sul C64 dovete predisporre il bit 2 del byte della forma d'onda quando si utilizza una forma d'onda triangolare (inserite il valore numerico 4 nel registro 21 con una istruzione POKE). La voce 3 deve essere predisposta ad una qualsiasi frequenza. Nessun altro parametro della voce 3 ha influenza sulla modulazione sonora.

Anche la sincronizzazione sul C64 somma due toni, in modo da ottenere un effetto sonoro nuovo e differente. Se il bit 1 della forma d'onda è posto a 1 (inserite il valore numerico 4 nel registro 19 con una POKE), predisponendo la voce 3 ad un tono definito (con una POKE ai registri 14 e 15 per il tono della voce 3) e manipolando il tono della voce 1 (registri 0 e 1) si provoca il cambiamento delle qualità timbriche del tono risultante.

La sincronizzazione si verifica quando le due forme d'onda sono collegate in modo da rendere la forma d'onda della voce 1 dipendente dal fatto di essere *sincronizzata* con la frequenza prodotta dalla voce 3. Dal momento che le due forme d'onda non sono generalmente in sincronia, la forma d'onda è distorta, producendo forme d'onda differenti ed a volte interessanti. In sincronia, il timbro del tono che udite dipende dal timbro della voce 3, non della voce 1, come accadrebbe normalmente.

Come maneggiare il chip SID

Il chip SID contiene anche due registri (25-26) connessi ai due

ingressi del joystick. Questi registri conterranno un numero compreso tra 0 e 255 dipendente dalla resistenza del potenziometro collegato all'ingresso (255 corrisponde alla massima resistenza). Dal momento che le paddle usate nei giochi sono realmente dei potenziometri (cioè delle resistenze variabili), questi ingressi possono essere utilizzati per registrare i movimenti delle paddle e possono essere facilmente usati per cambiare i valori in altri registri interni al chip mentre vengono prodotti dei suoni.

Questo semplicissimo sottoprogramma può essere aggiunto ad un programma sonoro per controllare il tono della voce 1 con una paddle inserita nell'ingresso 1 mentre viene prodotto un suono:

```
10 POKE 54272+1,PEEK(54272+25): GOTO 10
```

Questa riga collega il valore della paddle al byte più significativo del valore numerico che esprime la frequenza della voce 1. È molto più facile studiare gli effetti prodotti dai cambiamenti dei valori sonori, se potete ascoltare il suono che ne risulta mentre state facendo esperimenti. Questo è il principio di base di Compositore 2.

Compositore

Il programma Compositore 2 vi permette di creare effetti sonori personalizzati e di manipolarli cambiando diversi parametri. Attaccare, decadere, sostenere e rilasciare sono compresi, così come il tono, i filtri, la modulazione sonora e la sincronizzazione. La forma d'onda vibrata può essere manipolata in modo da cambiare l'ampiezza della vibrazione del suono, alterando in maniera considerevole il timbro del suono risultante.

Per utilizzare Compositore 2 copiatelo e registratelo su disco o su nastro. Quando siete sicuri di avere una copia registrata mandate in esecuzione il programma. Dopo un breve ritardo, nel corso del quale il programma carica in memoria un breve sottoprogramma in linguaggio macchina, il termine Attaccare appare nell'angolo superiore sinistro dello schermo. Usando i tasti + e -, potete aumentare o diminuire il valore di attaccare per il vostro effetto sonoro. Il valore che viene correntemente inserito in memoria da una istruzione POKE è rappresentato sia da un istogramma a barre sia da un valore numerico. Il valore numerico varia di quantità pari a 16 o ad 1 unità, in base al parametro su cui state lavorando. È previsto che questi valori servano solo come punto di riferimento, dal momento che

possono differire dal valore attuale di una unità in più o in meno. Gli aumenti sono stati scelti in modo da rendere molto più facile percepire il cambiamento avvenuto nei parametri e facilitare l'uso del programma.

Una volta che abbiate deciso circa il valore di attaccare premete semplicemente RETURN ed apparirà il parametro successivo. Ricordatevi che sostenere e volume devono essere valori numerici sufficientemente alti perché il suono sia percepibile. Quando avete scelto i valori da assegnare a ciascun parametro (l'ultimo ad apparire sullo schermo è quello che contraddistingue l'onda di vibrazione bassa) potete quindi produrre il suono con i tasti di funzione. Il tasto f1 produce l'effetto sonoro con una forma d'onda a dente di sega, f3 con la forma d'onda triangolare, f5 con la forma d'onda rumore e f7 con la forma d'onda vibrata.

La modulazione sonora e la sincronizzazione

Il tasto riportante la freccia verso l'alto (a fianco dell'asterisco) produce il suono come se fosse modulato con la voce 3 ed il tasto con la freccia verso sinistra (a fianco del tasto 1) genera l'effetto sonoro sincronizzato risultante dai timbri della voce 1 e della voce 3 (la modulazione sonora e la sincronizzazione si limitano alla voce 1).

Una volta che abbiate ascoltato la voce 1 limitatevi a premere il tasto 2 e vi verranno riproposti i parametri da modificare. Come per la voce 1, potete utilizzare la voce 2 con i tasti funzione. Per ascoltare le voci 1 e 2 simultaneamente premete la barra spaziatrice. Per scegliere i parametri della voce 3 premete il tasto 3. La barra spaziatrice suona quindi tutte le voci precedentemente definite. Se avete scelto la modulazione sonora o la sincronizzazione per la voce 1, non potrete utilizzare la voce 3 come suono separato.

Come si cambiano gli effetti sonori

Per cambiare un qualsiasi parametro in qualunque momento, dopo averlo inserito originariamente, limitatevi a premere il tasto che appare in negativo accanto al nome del parametro ed a premere i tasti + o - per aumentare o diminuire il valore associato. Quando avete finito battete RETURN.

Potete persino cambiare i parametri mentre il suono viene prodotto. Per farlo, premete uno dei tasti funzione o uno dei tasti

recanti le frecce per far partire la nota e, senza smettere di premerli, battete il carattere associato al parametro che volete cambiare. Quindi cambiate il suono con i tasti + e -.

Per usare i filtri mentre il suono viene emesso dovete innanzi tutto far partire l'effetto sonoro desiderato, quindi, senza lasciare i tasti, premere H (per passa-alto), B (per passa-banda) o L (per passa-basso). In seguito premete F per attivare il filtro ed usate i tasti + e - per incrementare o decrementare la frequenza di taglio. Come nel caso precedente, premete RETURN per porre fine alla nota.

Per registrare il suono o i suoni che avete creato premete Q mentre la nota viene emessa. Lo schermo viene cancellato e su di esso appare un programma. Battete NEW e premete RETURN in corrispondenza delle righe che vengono listate sullo schermo. Quindi potete produrre questo effetto sonoro, oppure registrarlo su nastro o su disco ed usarlo in seguito come sottoprograma nei vostri programmi. Per usarlo come sottoprogramma avrete bisogno di un ciclo di ritardo come il seguente per stabilire la durata:

```
70 FOR T=1 TO 2000:NEXT T
```

Quindi, per disattivare l'effetto sonoro usate la riga seguente:

```
80 FOR T=49152+4 TO 49152+18 STEP
   7:POKE T,(PEEK(T) AND 254):NE
  XT:SYS 53017
```

Per attivare il suono nei vostri programmi potete sia richiamare l'intero sottoprogramma con una istruzione GOSUB che usare la riga seguente (cui avrete attribuito il numero di riga opportuno):

```
FOR T=49152+4 TO 49152+18 STEP 7:
POKE T,(PEEK(T) OR 1):NEXT:SYS 53
017
```

Qualche nota riguardo al programma

Il programma Compositore 2 usa un breve sottoprogramma in linguaggio macchina, che copia il contenuto di 24 byte a partire dalla locazione 49152 nel registro sonoro che inizia in 54272. Il sottoprogramma in linguaggio macchina copia i registri nello stesso ordine in cui si dovrebbe accedere loro con delle istruzioni POKE, in modo da creare un effetto sonoro nella maniera più opportuna.

Ciò viene fatto perché i registri sonori sono registri a *sola scrittura*. Vale a dire, quando dei valori numerici vengono inseriti nei registri SID tramite delle istruzioni POKE, non possono essere successiva-

mente letti con delle PEEK. Dovete, invece, conservare i valori suddetti in variabili od altre locazioni di memoria. Il sottoprogramma in linguaggio macchina conserva questi valori in un'area di memoria sicura e ci consente di copiarli in qualsiasi momento nei registri SID. La capacità di ricordare i valori che sono stati inseriti tramite POKE nel chip SID rende possibile il programma Compositore 2.

```
100 I=52992
110 READ A:IF A=256 THEN 190
120 POKE I,A:I=I+1:GOTO 110
130 DATA 24,5,6,0,1,2,3
140 DATA 21,12,13,7,8,9,10
150 DATA 11,19,20,14,15,16,17
160 DATA 23,4,11,18,162,0,188
170 DATA 0,207,185,0,192,153,0
180 DATA 212,232,224,25,208,242,96,256
190 POKE53281,1:POKE53280,1
200 POKE650,128
210 F$="{ 19 SPAZI}"
220 S=49152:D=0:Q=54272:P=53017:M$="VOCE":
    Z$="{ 4 SPAZI}{ 4 SIN}"
230 FORT=STOS+30:POKET,0:NEXT:SYSP
240 PRINT"{CLR}";:FI$=" NESSUNA{ 3 SPAZI}
    "
250 FORA=1TO11:ON A GOSUB500,510,520,530,5
    40,550,560,570,590,600,610:NEXT
270 GETE$:U=PEEK(197):IFU=64ANDPEEK(S+4)TH
    ENPOKES+4,PEEK(S+4)AND254:SYSP
280 IFU=64ANDPEEK(S+7+4)THENPOKES+7+4,PEEK
    (S+7+4)AND254:SYSP
290 IFU=64ANDPEEK(S+14+4)THENPOKES+14+4,PE
    EK(S+14+4)AND254:SYSP
300 IFU=62THENSYSP:GOTO1330
310 IFE$="1"ORE$="2"ORE$="3"THEND=(ASC(E$)
    -49)*7:PRINT"{CLR}";TAB(25);M$;E$:GOTO
    250
320 IFD>7THENPOKES+24,(PEEK(S+24)AND127):S
    YSP
330 IFU=4THENPOKES+4+D,33:SYSP
```

```
340 IFU=5THENPOKES+4+D,17:SYSP
350 IFU=6THENPOKES+4+D,129:SYSP
360 IFU=3THENPOKES+4+D,65:SYSP
370 IFU=39THENPOKES+24,(PEEK(S+24)AND255):
    FI$=" NESSUNA{ 6 SPAZI}":POKES+23,0:SY
    SP
380 IF U=60 THENFORT=0TO14STEP7:POKES+4+T,
    PEEK(S+4+T)OR1:NEXT:SYSP
390 IFU=57THENPOKES+4+D,PEEK(S+4+D)OR3:SYS
    P
400 IFU=54THENPOKES+4+D,21:SYSP
410 V=2↑(D/7)
420 IFU=42THENFI$=" PASSA-BASSO ":POKES+23
    ,V:POKES+24,(PEEK(S+24)OR16):SYSP
430 IFU=29THENFI$=" PASSA-ALTO ":POKES+23,
    V:POKES+24,(PEEK(S+24)OR64):SYSP
440 IFU=28THENFI$=" PASSA-BANDA ":POKES+23
    ,V:POKES+24,(PEEK(S+24)OR32):SYSP
450 N$="ADSROYTVFPI":FORJ=1TO LEN(N$):G$=M
    ID$(N$,J):IF LEFT$(G$,1)=E$THEN480
460 NEXT
470 GOTO270
480 ONLEN(G$)GOSUB610,600,590,580,560,550,
    540,530,520,510,500
490 GOTO270
500 PRINT"{BLK}{HOME}TASSO DI {RVS}A{OFF}T
    TACCARE +-":GOSUB620:RETURN
510 PRINT"{BLU}{HOME}{ 2 GIU'}TASSO DI
    {RVS}D{OFF}ECADERE +-":GOSUB700:RETURN

520 PRINT"{RED}{HOME}{ 4 GIU'}LIVELLO DI
    {RVS}S{OFF}OSTENERE +-":GOSUB770:RETUR
    N
530 PRINT"{GRN}{HOME}{ 6 GIU'}TASSO DI
    {RVS}R{OFF}ILASCIARE +-":GOSUB840:RETU
    RN
540 PRINT"[<1>]{HOME}{ 8 GIU'}V{RVS}O{OFF}
    LUME GENERALE +-":GOSUB910:RETURN
550 PRINT"[<2>]{HOME}{ 10 GIU'}TONO (B
```

```
    {RVS}Y{OFF}TE +)+-":GOSUB970:RETURN
560 PRINT"{PUR}{HOME}{ 12 GIU'}{RVS}T{OFF}
    ONO (BYTE -)+-":GOSUB1030:RETURN
570 IFD>0THENPRINT"{HOME}{ 14 GIU'}NE' MOD
    ULAZIONE NE' SINCRONIZZAZIONE"
575 IFD>0THENPRINT"{HOME}{ 15 GIU'}PER LE
    VOCI DUE E TRE"
580 PRINT"[<7>]{HOME}{ 14 GIU'}TONO {RVS}V
    {OFF}OCE 3 (PER MODULAZIONE)+-":GOSUB1
    090:RETURN
590 PRINT"[<4>]{HOME}{ 16 GIU'}{RVS}F{OFF}
    REQUENZA DI TAGLIO DEI FILTRI+-":GOSUB
    1150:RETURN
600 PRINT"[<3>]{HOME}{ 18 GIU'}ONDA {RVS}P
    {OFF}ULSANTE SUP. +-":GOSUB1210:RETURN

610 PRINT"[<2>]{HOME}{ 20 GIU'}ONDA PULSAN
    TE {RVS}I{OFF}NF. +-":GOSUB1270:RETURN

620 POKE198,0:GETA$:IF A$<>""THEN620
630 IF PEEK(197)<>40ANDPEEK(197)<>43ANDPEE
    K(197)<>1THEN680
640 IFPEEK(197)=40ANDX1<15THENX1=X1+1
650 IFPEEK(197)=43ANDX1>0THENX1=X1-1
660 IFPEEK(197)=1THENPOKE197,0:POKE198,0:F
    ORT=1TO500:NEXT:POKE198,0:PRINT:RETURN

670 PRINT"{RVS}";LEFT$(F$,X1);"{OFF}";RIGH
    T$(F$,15-X1);Z$;(PEEK(S+D+5)AND240);"
    { 2 SU}"
680 POKES+D+5,(X1*16)+(PEEK(S+D+5)AND15):P
    OKEQ+D+5,(PEEK(S+D+5))
690 GOTO630
700 POKE198,0:IF PEEK(197)<>40ANDPEEK(197)
    <>43ANDPEEK(197)<>1THEN750
710 IFPEEK(197)=40ANDX2<15THENX2=X2+1
720 IFPEEK(197)=43ANDX2>0THENX2=X2-1
730 IFPEEK(197)=1THENPOKE197,0:POKE198,0:F
    ORT=1TO500:NEXT:POKE198,0:PRINT:RETURN
```

```
740 PRINT"{RVS}";LEFT$(F$,X2);"{OFF}";RIGH
    T$(F$,15-X2);Z$;(PEEK(S+D+5)AND15);"
    {SU}"
750 POKES+D+5,X2+(PEEK(S+D+5)AND240):POKEQ
    +D+5,PEEK(S+D+5)
760 GOTO700
770 POKE198,0:IF PEEK(197)<>40ANDPEEK(197)
    <>43ANDPEEK(197)<>1THEN820
780 IFPEEK(197)=40ANDX3<15THENX3=X3+1
790 IFPEEK(197)=43ANDX3>0THENX3=X3-1

800 IFPEEK(197)=1THENPOKE197,0:POKE198,0:F
    ORT=1TO500:NEXT:POKE198,0:PRINT:RETURN

810 PRINT"{RVS}";LEFT$(F$,X3);"{OFF}";RIGH
    T$(F$,15-X3);Z$;(PEEK(S+D+6)AND240);"
    {SU}"
820 POKES+D+6,(X3*16)+(PEEK(S+D+6)AND15):P
    OKEQ+D+6,PEEK(S+D+6)
830 GOTO770
840 POKE198,0:IF PEEK(197)<>40ANDPEEK(197)
    <>43ANDPEEK(197)<>1THEN890
850 IFPEEK(197)=40ANDX4<15THENX4=X4+1
860 IFPEEK(197)=43ANDX4>0THENX4=X4-1
870 IFPEEK(197)=1THENPOKE197,0:POKE198,0:F
    ORT=1TO500:NEXT:POKE198,0:PRINT:RETURN

880 PRINT"{RVS}";LEFT$(F$,X4);"{OFF}";RIGH
    T$(F$,15-X4);Z$;(PEEK(S+D+6)AND15);"
    {SU}"
890 POKES+D+6,X4+(PEEK(S+D+6)AND240):POKEQ
    +D+6,PEEK(S+D+6)
900 GOTO840
910 POKE198,0:IF PEEK(197)<>40ANDPEEK(197)
    <>43ANDPEEK(197)<>1THEN960
920 IFPEEK(197)=40ANDX5<15THENX5=X5+1
930 IFPEEK(197)=43ANDX5>0THENX5=X5-1
940 IFPEEK(197)=1THENPOKE197,0:POKE198,0:F
    ORT=1TO500:NEXT:POKE198,0:PRINT:RETURN
```

```
950  PRINT"{RVS}";LEFT$(F$,X5);"{OFF}";RIGH
     T$(F$,15-X5);Z$;(PEEK(S+24)AND15);"
     {SU}"
960  POKES+24,(X5+(PEEK(S+24)AND240)):SYSP:
     GOTO910
970  POKE198,0:IF PEEK(197)<>40ANDPEEK(197)
     <>43ANDPEEK(197)<>1THEN1020
980  IFPEEK(197)=40ANDX6<15THENX6=X6+1
990  IFPEEK(197)=43ANDX6>0THENX6=X6-1
1000 IFPEEK(197)=1THENPOKE197,0:POKE198,0:
     FORT=1TO500:NEXT:POKE198,0:PRINT:RETU
     RN
1010 PRINT"{RVS}";LEFT$(F$,X6);"{OFF}";RIG
     HT$(F$,15-X6);Z$;PEEK(S+D+1);"{SU}"
1020 POKES+1+D,16*X6:POKEQ+1+D,PEEK(S+1+D)
     :GOTO970
1030 POKE198,0:IF PEEK(197)<>40ANDPEEK(197
     )<>43ANDPEEK(197)<>1THEN1080
1040 IFPEEK(197)=40ANDX7<15THENX7=X7+1
1050 IFPEEK(197)=43ANDX7>0THENX7=X7-1
1060 IFPEEK(197)=1THENPOKE197,0:POKE198,0:
     FORT=1TO500:NEXT:POKE198,0:PRINT:RETU
     RN
1070 PRINT"{RVS}";LEFT$(F$,X7);"{OFF}";RIG
     HT$(F$,15-X7);Z$;PEEK(S+D);"{SU}"
1080 POKES+D,16*X7:POKEQ+D,PEEK(S+D):GOTO1
     030
1090 POKE198,0:IF PEEK(197)<>40ANDPEEK(197
     )<>43ANDPEEK(197)<>1THEN1140
1100 IFPEEK(197)=40ANDX8<15THENX8=X8+1
1110 IFPEEK(197)=43ANDX8>0THENX8=X8-1
1120 IFPEEK(197)=1THENPOKE197,0:POKE198,0:
     FORT=1TO500:NEXT:POKE198,0:PRINT:RETU
     RN
1130 PRINT"{RVS}";LEFT$(F$,X8);"{OFF}";RIG
     HT$(F$,15-X8);Z$;PEEK(S+15+D);"{SU}"
1140 POKEQ+24,PEEK(S+24)OR128:POKES+15+D,X
     8*16:POKEQ+15+D,X8*16:GOTO1090
1150 POKE198,0:IF PEEK(197)<>40ANDPEEK(197
```

```
     )<>43ANDPEEK(197)<>1THEN1200
1160 IFPEEK(197)=40ANDX9<15THENX9=X9+1
1170 IFPEEK(197)=43ANDX9>0THENX9=X9-1
1180 IFPEEK(197)=1THENPOKE197,0:POKE198,0:
     FORT=1TO500:NEXT:POKE198,0:PRINT:RETU
     RN
1190 PRINT"{RVS}";LEFT$(F$,X9);"{OFF}";RIG
     HT$(F$,15-X9);Z$;PEEK(S+22);"
     { 6 DES}";FI$;"{SU}"
1200 POKES+21,X9/2:POKES+22,(X9*16):POKEQ+
     21,7:POKEQ+22,(X9*16):GOTO1150
1210 POKE198,0:IF PEEK(197)<>40ANDPEEK(197
     )<>43ANDPEEK(197)<>1THEN1260
1220 IFPEEK(197)=40ANDXA<15THENXA=XA+1
1230 IFPEEK(197)=43ANDXA>0THENXA=XA-1
1240 IFPEEK(197)=1THENPOKE197,0:POKE198,0:
     FORT=1TO500:NEXT:POKE198,0:PRINT:RETU
     RN
1250 PRINT"{RVS}";LEFT$(F$,XA);"{OFF}";RIG
     HT$(F$,15-XA);Z$;PEEK(S+D+2);"{SU}"
1260 POKES+D+2,XA*16:POKEQ+D+2,PEEK(S+D+2)
     :GOTO1210
1270 POKE198,0:IF PEEK(197)<>40ANDPEEK(197
     )<>43ANDPEEK(197)<>1THEN1320
1280 IFPEEK(197)=40ANDXB<15THENXB=XB+1
1290 IFPEEK(197)=43ANDXB>0THENXB=XB-1
1300 IFPEEK(197)=1THENPOKE197,0:POKE198,0:
     FORT=1TO500:NEXT:POKE198,0:PRINT:RETU
     RN
1310 PRINT"{RVS}";LEFT$(F$,XB);"{OFF}";RIG
     HT$(F$,15-XB);Z$;PEEK(S+D+3);"{SU}"
1320 POKES+D+3,XB*16:GOTO1270
1330 REM SAVE ROUTINE
1340 S=49152:CO=52992
1350 PRINT"{CLR}":DIMQ(45),ML(45)
1360 FORT=0TO44:Q(T)=PEEK(S+T):ML(T)=PEEK(
     CO+T):NEXT
1370 PRINT"1 RP=52992:FORR=RPTORP+44:READG
     P:POKER,GP:NEXT"
```

```
1380 PG=0:FORA=0TO4:PG=PG+3
1390 PRINT PG"DATA";:FORT=0TO8 :PRINTML(T+
     9*A);:IF T<8 THENPRINT"{SIN},";
1400 NEXT:PRINT:NEXT
1410 PRINT"20S=49152:FORT=STOS+24:POKET,0:
     NEXT:P=53017{ 2 SPAZI}"
1420 PRINT"30FORT=STOS+25:READDS:POKET,DS:
     NEXT:SYSP{ 3 SPAZI}"
1430 PO=30:FORW=0TO2:PO=PO+10
1440 PRINTPO"DATA";:FORT=0TO8:PRINTQ(T+9*W
     );:IFT<8THENPRINT"{SIN},";
1450 NEXT:PRINT:NEXT
```

Figura 1. Filtro passa-basso (la frequenza di taglio viene incrementata col tempo)

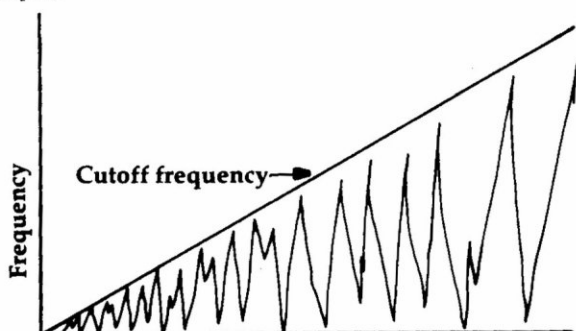



Figura 2. Filtro passa-banda (la frequenza di taglio viene incrementata col tempo)

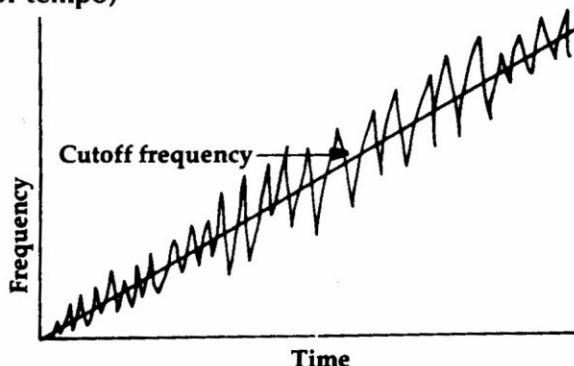
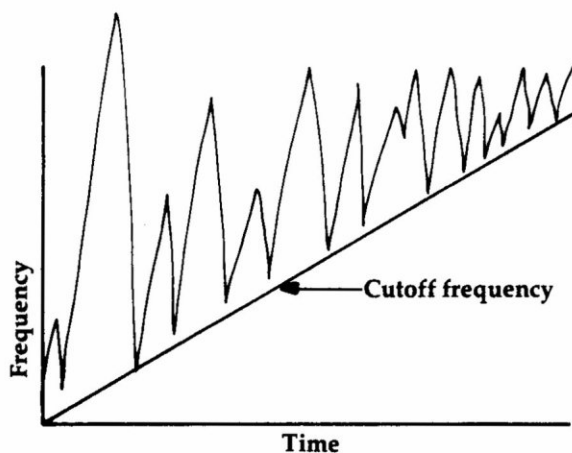



Figura 3. Filtro passa-alto (la frequenza di taglio viene incrementata col tempo)

Capitolo quattro

Musica

Compositore

W. J. Crowley

Comporre motivi con il Commodore 64 usando le istruzioni DATA può essere un procedimento piuttosto lungo. «Compositore» facilita il procedimento creando un file su disco o su nastro, evitando così di dover far uso delle istruzioni DATA. Le istruzioni per l'uso del programma sono contenute nel programma stesso.

Da orgoglioso neo possessore di un Commodore 64, sono stato immediatamente affascinato dalle capacità del dispositivo di interfaccia sonora (Sound Interface Device, SID). Dopo aver esplorato tutti gli esempi contenuti nella *Guida di riferimento del Programmatore* ho tentato di convertire uno spartito musicale in istruzioni DATA, in modo da mettere alla prova le capacità del SID con qualcuna delle mie canzoni e dei miei motivi preferiti.

Ho quindi pensato che sarebbe stato molto più conveniente avere un programma che effettuasse la traduzione da musica a dati e che fornisse un modo di registrare e ricuperare motivi da nastro o da disco, e quindi suonarli nuovamente. Il risultato di questo esperimento è il programma «Compositore».

Il programma

Facendo riferimento al listato del programma, le righe da 100 a 280 provvedono a predisporre lo schermo e forniscono diverse funzioni disponibili. Il sottoprogramma alle righe da 1000 a 1130 inizializza le variabili e predispone il dispositivo di interfaccia sonora nel modo descritto dalla *Guida di riferimento del Programmatore,* a parte il fatto che ho cambiato le matrici contenenti le note dalla notazione in virgola mobile a quella intera. Dal momento che le variabili intere richiedono meno memoria, c'è spazio in memoria per motivi più lunghi.

Il sottoprogramma di AIUTO inizia alla riga 1500 e dà una sintetica spiegazione di come si usa il programma «Compositore». La riga 1510 cambia l'uscita video da maiuscola in minuscola, per facilitare la lettura. Le righe 1660 e 1670 interrompono la visualizzazione sullo schermo fino a che non premete la barra spaziatrice. Le righe da 1700 a 1790 forniscono un esempio dell'aspetto che avrà lo

schermo quando inizierete ad inserire le note. La riga 1830 riporta l'uscita del video in formato maiuscolo/grafico.

Le righe da 5000 a 6170 rappresentano il modulo di preparazione/correzione del motivo. La riga 5030 inizia accettando le note destinate alla voce 1 e crea un ciclo per accettare le note per le voci 2 e 3. La riga 5090 verifica se avete battuto un numero negativo, indicante una pausa. La riga 6000 interpreta una eventuale risposta 0 come desiderio di passare alla voce successiva (o di ritornare al menu principale, se avete completato la terza voce). Le righe 6030 e 6040 esaminano se volete spostare le note di un'ottava verso l'alto o verso il basso e quindi rivisualizzare l'informazione, se avete operato un cambiamento. La riga 6050 converte il nome della nota normalmente usato nella notazione musicale in un numero equivalente. Le righe da 6060 a 6150 sono ricavate direttamente dalla *Guida di riferimento del Programmatore*, e la riga 6170 ritorna il controllo al menu principale. Le righe da 9000 a 9180 operano la conversione da nome della nota a numero. Notate che, se viene battuto qualche nome non valido, questo modulo assume per difetto la nota più bassa (Do, per l'ottava corrente).

Le righe da 10000 a 10150 si occupano di caricare un motivo da nastro. Gli utenti di unità dischi dovrebbero apportare alle righe 10040 e 20040 i cambiamenti necessari, indicati dalle istruzioni REM.

Le righe da 15000 a 15030 predispongono lo schermo per suonare nuovamente i motivi ed acquisiscono da tastiera un numero per il tempo prima che il motivo venga suonato. Ho lasciato dello spazio tra le righe 15030 e 15100, pensando che avrei in seguito potuto desiderare di acquisire in questa sezione qualche forma di controllo delle forme d'onda. Anche le righe dalla 15100 alla 15200 sono ricavate quasi letteralmente dalla *Guida di riferimento del Programmatore*, che ne spiega molto chiaramente la funzione.

Ora copiamo il programma, in modo da vedere come funziona. Dopo averne registrato una copia su nastro mandatelo in esecuzione con una RUN e osservate il menu dei comandi mostrato sullo schermo. La prima scelta proposta è AIUTO, ma invece di esaminarla per prima facciamo un semplice esempio. Battete P (senza premere RETURN) e notate come si passi al modulo di preparazione/correzione dei motivi, in cui il programma chiede di indicare un titolo per il motivo. Battete JUNQUE o qualche altro titolo fittizio; notate come si stiano introducendo i dati per la voce 1. "Compo-

sitore" richiede la *durata*, cioè quanto a lungo deve durare la nota. Rispondete come segue:

Il programma domanda	**Voi rispondete**
DURATA?	0 (zero)
DATI PER LA VOCE N. 2	
DURATA?	4
OTTAVA = 4:NOTA?	-(simbolo meno)
OTTAVA = 3:NOTA?	SI-(SI bemolle)
DURATA?	2
OTTAVA = 3:NOTA?	+(simbolo più)
OTTAVA = 4:NOTA?	FA
DURATA?	4
OTTAVA = 4:NOTA?	-
OTTAVA = 3:NOTA?	FA
DURATA?	2
OTTAVA = 3:NOTA?	FA
DURATA?	4
OTTAVA = 3:NOTA?	+
OTTAVA = 4:NOTA?	FA
DURATA?	

Ora osserviamo lo schermo. Notate che, a parte quando vi trovate all'interno del menu principale, tutte le vostre risposte sono seguite da RETURN. Le risposte a DURATA? sono sempre rappresentate da un numero (numeri negativi per le pause). Alle domande riguardanti le note si risponde sempre con un più (+), un meno (-) od il nome di una nota (da DO a SI).

Il programma si trova in attesa su di una domanda DURATA?: rispondete quindi con uno zero (0) e notate come si passi alla voce 3. Rispondete anche a questa domanda circa la durata con uno zero (0), in modo da ritornare al menu dei comandi. Battendo E si passa al modulo di esecuzione ed il titolo del motivo verrà visualizzato. Il TEMPO del motivo è predisposto per difetto ad 80, quindi limitatevi ad alzare il volume del vostro monitor e battete RETURN. Sentirete suonare il nostro breve motivo e quindi si ritorna al menu principale. Ora scegliamo di nuovo la sezione E, ma questa volta

rispondiamo alla domanda circa il TEMPO con un valore numerico di 40 e notate la differenza nella velocità con cui le note vengono prodotte. Se suonate nuovamente questo stesso motivo, vi accorgerete che questo TEMPO (40) viene mantenuto fino a che lo cambiate nuovamente. Se ora scegliete il modulo di preparazione/correzione dei motivi, potete inserire informazioni per le voci 1 e 3, senza cambiare od interferire con la voce 2. Potete anche modificare la voce 2, se volete. Ora può darsi vogliate provare a registrare e recuperare esempi da nastro, per completare il test del programma.

Se volete provare a comporre una canzone di vostra scelta, battete l'opzione F internamente al menu principale, per interrompere il programma, quindi battete RUN, in modo da azzerare tutte le variabili precedentemente usate.

Compositore

```
100 GOSUB1000:REM INIZIALIZZAZIONE
110 REM ***** SELEZIONE COMANDI *****
120 GOSUB1400
130 PRINT"{ 6 SPAZI}MENU' COMANDI.........
    ."
140 PRINT"{ 6 SPAZI}A...AIUTO/ISTRUZIONI"
150 PRINT"{ 6 SPAZI}P...PREPARAZIONE/CORRE
    ZIONE"
155 PRINT"{ 10 SPAZI}DEL MOTIVO"
160 PRINT"{ 6 SPAZI}L...LETTURA DEL MOTIVO
     DA"
165 PRINT"{ 10 SPAZI}CASSETTA"
170 PRINT"{ 6 SPAZI}E...ESECUZIONE DEL MOT
    IVO"
180 PRINT"{ 6 SPAZI}R...REGISTRAZIONE DEL
    MOTIVO"
185 PRINT"{ 10 SPAZI}SU CASSETTA"
190 PRINT"{ 6 SPAZI}F...FINE PROGRAMMA"
200 PRINT"{ 6 SPAZI}COMANDO ?"
210 GETI$:IFI$=""THEN210
220 IFLEFT$(I$,1)="A"THEN1500
230 IFLEFT$(I$,1)="P"THEN5000
240 IFLEFT$(I$,1)="L"THEN10000
```

```
250 IFLEFT$(I$,1)="E"THEN15000
260 IFLEFT$(I$,1)="R"THEN20000
270 IFLEFT$(I$,1)="F"THEN999
280 GOTO120
999 GOSUB1400:PRINT".....FINE":END
1000 REM **** INIZIALIZZAZIONE ****
1010 S=54272:FORL=STOS+24:POKEL,0:NEXT
1020 DIMH%(2,600),L%(2,600),C%(2,600)
1030 DIMFQ(11)
1040 V(0)=17:V(1)=65:V(2)=33
1060 FORI=0TO11:READFQ(I):NEXT
1070 TEMPO=80:OCT%=4
1080 RETURN
1100 REM ---- DATI OTTAVA SUPERIORE ----
1110 DATA34334,36376,38539,40830
1120 DATA43258,45830,48556,51443
1130 DATA54502,57743,61176,64814
1400 REM --- ROUTINE PULIZIA SCHERMO ---
1410 PRINT"{CLR}":FORI=1TO8:PRINTCHR$(13);
     :NEXT
1420 RETURN
1500 REM **** AIUTO/ISTRUZIONI ****
1510 GOSUB1400:PRINTCHR$(14)
1520 PRINT"PER COMPORRE UN MOTIVO, BATTETE
      LA DURA"
1525 PRINT"TA L'OTTAVA, E IL NOME DI OGNI
     NOTA PER"
1530 PRINT"CIASCUNA DELLE TRE VOCI."
1540 PRINT"LA DURATA E' ESPRESSA IN 16-ESI
     MI, QUIN-DI 8=8/16, O MEZZA NOTA."
1550 PRINT"UNA RISPOSTA '0' SIGNIFICA NESS
     UN ALTRA"
1555 PRINT"NOTA PER QUESTA VOCE; UN NUMERO
      NEGATI-"
1560 PRINT"VO SIGNIFICA UNA PAUSA; -4=-4/1
     6-ESIMI "
1570 PRINT"O UNA PAUSA DI UN QUARTO.":PRIN
     T
1580 PRINT"LE NOTE VENGONO INSERITE BATTEN
```

```
     DONE IL"
1585 PRINT"NOME (DA DO A SI). IL NOME SEGU
     ITO DA"
1590 PRINT"'+' SIGNIFICA DIESIS, '-' SIGNI
     FICA BE-"
1600 PRINT"MOLLE. QUINDI SOL+ SIGNIFICA S
     OL DIESIS"
1610 PRINT"E SOL- SIGNIFICA SOL BEMOLLE.":
     PRINT
1620 PRINT"SE RISPONDETE ALLA DOMANDA CIRC
     A LA NO-"
1625 PRINT"TA CON '+' MA NESSUN NOME, L'OT
     TAVA VER"
1630 PRINT"RA' AUMENTATA (PIU' ALTA); UN '
     -' (DA"
1640 PRINT"SOLO) FARA' DIMINUIRE L'OTTAVA.
     "
1650 PRINT"...............................
     ........":PRINT
1660 PRINT"PREMERE{SPAZI}BA{ 2 R}A{SPAZI}
     SPAZIATRICE{SPAZI}PER{SPAZI}CONTINUA
     RE"
1670 GETA$:IFA$<>" "GOTO1670
1680 GOSUB1400
1690 PRINT"E{ 2 C}O{SPAZI}UN{SPAZI}ESEMPI
     O{SPAZI}DI{SPAZI}COME{SPAZI}PUO'
     {SPAZI}A{ 2 P}ARIRE{SPAZI}"
1700 PRINT"LO{SPAZI}SCHERMO:":PRINT
1710 PRINT"{ 5 SPAZI}L'USCITA{ 13 SPAZI}SI
     GNIFICA":PRINT
1720 PRINT"{ 40 *}":PRINT
1730 PRINT"{ 4 SPAZI}DURATA?{ 2 SPAZI}8
     { 10 SPAZI}(1/2 NOTA)"
1740 PRINT"O{ 2 T}AVA=4{ 5 SPAZI}NOTA?
     { 2 SPAZI}+{ 3 SPAZI}(UN OTTAVA SU)"
1750 PRINT"O{ 2 T}AVA=5{ 5 SPAZI}NOTA?
     { 2 SPAZI}DO"
1760 PRINT"{ 4 SPAZI}DURATA?{ 2 SPAZI}-1
     { 9 SPAZI}(PAUSA DI 1/16)"
```

```
1770 PRINT"{ 4 SPAZI}DURATA?{ 2 SPAZI}4
     { 10 SPAZI}(1/4 DI NOTA)"
1780 PRINT"O{ 2 T}AVA=5{ 5 SPAZI}NOTA?
     { 2 SPAZI}FA+ (FA DIESIS)"
1790 PRINT"{ 4 SPAZI}DURATA?{ 2 SPAZI}0
     { 10 SPAZI}(FINE VOCE)":PRINT
1800 PRINT"PREMERE{SPAZI}BA{ 2 R}A{SPAZI}
     SPAZIATRICE{SPAZI}PER{SPAZI}TORNARE"
1810 PRINT"AL{SPAZI}MENU"
1820 GETA$:IFA$<>" "THEN1820
1830 PRINTCHR$(142):GOTO110
5000 REM **** PREPARAZIONE/CORREZIONE DEL
     MOTIVO ****
5010 GOSUB1400:REM - CANCELLAZIONE SCHERMO

5020 PRINTTAB(10);"PREPARAZIONE MOTIVO":PR
     INT:INPUT"{ 4 SPAZI}TITOLO";F$
5030 FORK=0TO2
5040 PRINT:PRINT"{ 4 SPAZI}DATI PER LA VOC
     E N.";K+1
5050 PRINT
5060 I=0
5070 N$="":NT=0:DUR%=0
5080 INPUT"{ 8 SPAZI}DURATA";DUR%
5090 IFDUR%<0THEN6060:REM= UNA PAUSA
6000 IFDUR%=0THEN6150:REM= VOCE SUCCESSIVA

6010 PRINT"{ 3 SPAZI}OTTAVA = ";OCT%;
6020 INPUT" :{ 3 SPAZI}NOTA{ 3 SPAZI}";N$
6030 IFLEFT$(N$,1)="+"THENOCT%=OCT%+1:GOTO
     6010
6040 IFLEFT$(N$,1)="-"THENOCT%=OCT%-1:GOTO
     6010
6050 GOSUB9000:REM- CAMBIA IL NOME DELLA N
     OTA CON IL NUMERO CORRISPONDENTE
6060 WA=V(K):IFDUR%<0THENDUR%=-DUR%:WA=1
6070 FR=FQ(N)
6080 IFOCT%=7THEN6100
6090 FORJ=6TOOCT%STEP-1:FR=FR/2:NEXT
```

```
6100 HF%=FR/256:LF%=FR-256*HF%
6110 IFDUR%=1THENH%(K,I)=HF%:L%(K,I)=LF%:C
     %(K,I)=WA:I=I+1:GOTO5070
6120 FORJ=1TODUR%-1:H%(K,I)=HF%:L%(K,I)=LF
     %:C%(K,I)=WA:I=I+1:NEXT
6130 H%(K,I)=HF%:L%(K,I)=LF%:C%(K,I)=WA-1
6140 I=I+1:GOTO5070
6150 IFI>IMTHENIM=I
6160 NEXTK
6170 GOTO110
9000 REM ---- CAMBIA IL NOME DELLA NOTA NE
     L NUMERO CORRISPONDENTE
9010 IFN$="DO"THENN=0:RETURN
9020 IFN$="DO+"THENN=1:RETURN
9030 IFN$="RE-"THENN=1:RETURN
9040 IFN$="RE"THENN=2:RETURN
9050 IFN$="RE+"THENN=3:RETURN
9060 IFN$="MI-"THENN=3:RETURN
9070 IFN$="MI"THENN=4:RETURN
9080 IFN$="FA"THENN=5:RETURN
9090 IFN$="FA+"THENN=6:RETURN
9100 IFN$="SOL-"THENN=6:RETURN
9110 IFN$="SOL"THENN=7:RETURN
9120 IFN$="SOL+"THENN=8:RETURN
9130 IFN$="LA-"THENN=8:RETURN
9140 IFN$="LA"THENN=9:RETURN
9150 IFN$="LA+"THENN=10:RETURN
9160 IFN$="SI-"THENN=10:RETURN
9170 IFN$="SI"THENN=11:RETURN
9180 RETURN
10000 REM  CARICAMENTO MOTIVO - NASTRO
10010 GOSUB1400
10020 PRINTTAB(10);"CARICAMENTO MOTIVO DA
      NASTRO":PRINT
10030 INPUT"TITOLO ";F$
10040 OPEN1,1,0,F$:REM PER GLI UTENTI DI U
      NITA' DISCHI, OPEN 1,8,2,F$
10050 INPUT#1,IM
10060 FORK=0TO2
```

```
10070 I=0
10100 INPUT#1,L%(K,I),H%(K,I),C%(K,I)
10110 IFL%(K,I)=0THEN10130
10120 I=I+1:GOTO10100
10130 NEXTK
10140 CLOSE1
10150 GOTO110
15000 REM **** ESECUZIONE MOTIVO ****
15010 GOSUB1400
15020 PRINT"{ 9 SPAZI}ESECUZIONE ";F$:PRIN
      T
15030 INPUT"{ 4 SPAZI}TEMPO (NORM=80)";TEM
      PO
15100 POKES+10,8:POKES+22,128:POKES+23,244

15110 POKES+5,0:POKES+6,240
15120 POKES+12,85:POKES+13,133
15130 POKES+19,10:POKES+20,197
15140 POKES+24,31
15150 FORI=0TOIM
15160 POKES,L%(0,I):POKES+7,L%(1,I):POKES+
      14,L%(2,I)
15170 POKES+1,H%(0,I):POKES+8,H%(1,I):POKE
      S+15,H%(2,I)
15180 POKES+4,C%(0,I):POKES+11,C%(1,I):POK
      ES+18,C%(2,I)
15190 FORT=1TOTEMPO:NEXT:NEXT
15200 FORT=1TO200:NEXT:POKES+24,0
15210 GOTO110
20000 REM **** REGISTRAZIONE MOTIVO - NAST
      RO
20010 GOSUB1400
20020 PRINTTAB(10);"REGISTRAZIONE ";F$:PRI
      NT
20030 C$=","
20040 OPEN1,1,2,F$:REM PER GLI UTENTI DI U
      NITA' DISCHI, OPEN 1,8,2,F$+",S,W"
20050 PRINT#1,IM
20060 FORK=0TO2
```

```
20070 I=0
20100 PRINT#1,L%(K,I)C$H%(K,I)C$C%(K,I)
20110 IFL%(K,I)=0THEN20130
20120 I=I+1:GOTO20100
20130 NEXTK
20140 CLOSE1
20150 GOTO110
```

Capitolo quattro

Concertista

Chris Metcalf e Marc Sugiyama

Questo magnifico programma simula in tempo reale un pannello di controllo da sintetizzatore, con tutte le funzioni desiderabili per realizzare effetti sonori e musica con il Commodore 64. La vostra tastiera diventa il collegamento tra voi ed i suoni che udite. Lo schermo mostra una doppia tastiera da pianoforte e lo stato degli altri elementi che definiscono il suono che state generando.

Le funzioni di «Concertista» comprendono: melodia, accesso mediante pressione di un solo tasto agli accordi principali, timbro, involuzione, durata, ottava, mantenere, polifonia, forme d'onda ed altre. Tutte disponibili immediatamente ed automaticamente da tastiera.

La potenza e versatilità del «sintetizzatore musicale su di un solo chip» del C64 offrono, al musicista-programmatore, straordinarie capacità di controllo sul suono: la sua forma, il colore, persino l'interazione tra i suoni (modulazione). Esiste un'ampia libertà di scelta, ma ciò significa anche che ci sono molti aspetti di ciascun suono che il programmatore deve controllare. "Concertista" automatizza questo controllo: ad esempio, potete suonare accordi altrettanto facilmente di singole note. Oltre tutto imparerete a conoscere il significato dei vari registri sonori, poiché sentirete gli effetti provocati dal cambiamento dei registri. Ora potete cominciare ad esplorare a fondo i fantastici effetti sonori del C64.

Copiate il programma «Concertista» nello stesso modo in cui copiereste un qualsiasi programma BASIC. "Concertista" comprende due brevi sottoprogrammi in linguaggio macchina contenuti in istruzioni DATA, quindi assicuratevi che quei numeri siano stati copiati correttamente. Dopo aver copiato e registrato il programma mandatelo in esecuzione. Assicuratevi che il volume del vostro televisore o del dispositivo di uscita audio sia abbastanza alto da poter sentire il calcolatore.

Poco dopo che il messaggio ATTENDERE PREGO è scomparso dallo schermo il calcolatore mostrerà le istruzioni. Sulla riga superiore dello schermo troverete una fila di indicatori. Il primo termine della riga è OTTAVA, che può variare da 1 a 8. Questo termine è

seguito dal numero che contraddistingue la VOCE e che indica il particolare *timbro* di uscita. Di seguito troviamo una serie di lettere, che indicano il formato in cui attualmente si opera. Questi formati verranno descritti più avanti. L'ultima indicazione riguarda il VOLUME, che ha un campo di variabilità da 1 a 15.

La doppia tastiera

Sotto la riga contenente le indicazioni si trovano le due tastiere musicali. Indicano i tasti da premere sul calcolatore. La tastiera inferiore è la continuazione della superiore; quindi la serie di tasti inferiore produce le note più alte.

Al di sotto delle tastiere troviamo una descrizione delle funzioni assegnate ai tasti di funzione programmabili. La colonna di sinistra descrive ciò che si può ottenere premendo i soli tasti di funzione senza premere SHIFT e la colonna di destra le funzioni ottenibili premendo simultaneamente SHIFT.

f1 e **f3**: questi tasti vi consentono di cambiare il volume del suono. Premendo f1 aumenterete il volume di una unità; e premendo f3 otterrete una diminuzione della stessa entità. Notate come l'indicazione del VOLUME cambi quando premete uno o l'altro dei due tasti. Ricordatevi che il volume varia da 0 a 15; 0 è completamente spento; 15 è il volume massimo ottenibile.

f4: premendo f4 cambierà lo stato del formato Mantenere, indicato dalla seconda M della fila di indicatori. Quando questo formato è operativo, la M apparirà in negativo. Quando questo formato è attivato, il calcolatore non rilascerà il motivo dopo che il tasto è stato premuto. Il suono, invece, continua fino a che vengono premuti altri tasti. Per disattivare tutte le voci premete la barra spaziatrice.

f6: questo tasto cambia lo stato del formato Multivoce. Questo formato è indicato dalla V nella riga degli indicatori. Una V in negativo indica che il formato è operativo. Il formato Multivoce consente di usare più di una voce contemporaneamente. Il programma viene potenziato quando questo formato viene attivato. Se questo formato non è attivato, ciascuna nota segue la precedente sulla stessa voce e non si possono suonare degli accordi. Questo porta alcuni svantaggi, ma risulta utile accoppiato con il formato Melodia. Con questo formato potete avere fino a tre voci contemporaneamente.

f7 e **F5**: premendo questi due tasti si cambia lo stato dei formati

Melodia e Accordo. Verranno descritti più avanti.
f2: questo tasto permette di definire delle forme d'onda personalizzate.

Come produrre musica

Una volta che il programma è pronto premete questa sequenza di tasti: QWERTYUI. Dovreste sentire una scala di DO maggiore. Se non è così, controllate il programma in cerca di errori di battitura. Ora provate questa sequenza di tasti: IOP@*^ (RUN/STOP) Z. Questa volta sentirete la stessa scala, solo una ottava più alta.

Premendo la sequenza ZXCVBNM si ottiene un'altra scala, una ottava superiore della precedente. Ora provate a premere i tasti QET tutti insieme per ottenere un accordo di DO maggiore. Ciascuna nota di questo accordo è assegnata ad una voce. Dal momento che sono disponibili solo tre voci, il calcolatore è in grado di accettare in ingresso solo tre tasti per volta.

Se volete cambiare ottava, premete il tasto CTRL ed un numero da 1 a 8, essendo 1 l'ottava più bassa ed 8 la più alta. Alcune delle voci non funzionano molto bene nelle ottave molto basse. Premendo il tasto Commodore ed un numero cambierà il numero della VOCE. Anche questa ha un campo di variabilità che va da 1 a 8.

Il formato Melodia è molto interessante. Una M in negativo al secondo posto della riga di stato indica che il formato Melodia è attivo. Questo formato funzionerà indipendentemente dai formati Multivoce e Mantenere. Quando è in questo formato, il calcolatore passa dolcemente da una nota all'altra, invece di spostarsi di mezze note come farebbe un pianoforte. Questo può produrre un effetto piuttisto sconcertante, misterioso quando anche il formato Mantenere è attivo. Ad esempio, inserite il formato Melodia, assicuratevi che i formati Multivoce e Mantenere siano attivati e premete la seguente sequenza di tasti: QETIP*ZCB. Come al solito potete azzerare tutte le voci premendo la barra spaziatrice.

Come formare degli accordi

Un altro formato operativo è il formato Accordo. Questo formato vi permette di avere un controllo con un *solo tasto* su differenti tipi di accordi e sulla loro trasposizione. Una volta che avete attivato il formato Accordo apparirà una seconda riga di indicatori. Sulla sinistra compare il nome dell'accordo e sulla destra la sua posizione:

base (fondamentale), prima o seconda trasposizione.

L'accordo di base è un accordo in cui la nota più bassa rappresenta anche la chiave dell'accordo. Ad esempio, la triade armonica DO maggiore è formata usando le note DO, MI e SOL.
Quando le note sono in quest'ordine, DO-MI-SOL, l'accordo è un accordo di base. Se le note dell'accordo partono con una nota differente dal DO, allora abbiamo la trasposizione dell'accordo. Ad esempio, MI e SOL, con DO acuto, rappresentano la prima trasposizione e SOL, con MI e DO acuti, la seconda.

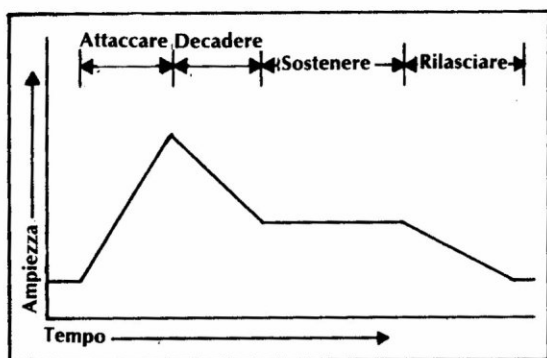
Per cambiare il tipo di accordo premete il tasto SHIFT ed un numero da 1 a 9. Gli accordi disponibili corrispondono ai seguenti numeri: (1) Maggiore; (2) Minore; (3) Diminuito; (4) Aumentato; (5) 7. maggiore; (6) 7. minore; (7) 7. dominante; (8) 6. maggiore; (9) 6. minore.

Le trasposizioni vengono selezionate premendo il tasto SHIFT ed il segno più per l'accordo fondamentale, il segno meno per la prima trasposizione ed il simbolo della lira per la seconda trasposizione.

Per poter suonare un accordo dovete innanzi tutto scegliere il tipo di accordo e di trasposizione desiderato e quindi premere sulla tastiera la nota corrispondente alla nota più bassa del vostro accordo. Ad esempio, se volete suonare un accordo in SI bemolle minore seconda trasposizione, ponetevi nel formato Accordo, scegliete l'accordo minore e la seconda trasposizione (premendo SHIFT-2 e SHIFT-£ premete R, che corrisponde alla nota FA sulla tastiera musicale. L'accordo che udrete è composto dalle seguenti note: FA, SI bemolle e RE diesis acuto (dal momento che il formato Melodia può armonizzare una sola voce per volta, i formati Accordo e Melodia sono incompatibili, quindi attivarne uno disattiva automaticamente l'altro).

Attaccare, decadere, sostenere, rilasciare

Per definire delle forme d'onda personalizzate premete f2. Una volta che vi siete posti in questo formato il calcolatore pone una serie di domande riguardanti la definizione di una nuova forma d'onda. La prima domanda riguarda quale voce desiderate cambiare. Premendo RETURN, senza alcun'altra risposta, si ritorna il controllo del programma al formato normale. Dopo questa domanda il calcolatore visualizza i valori correnti di attaccare, decadere, sostenere e rilasciare e richiede i nuovi valori. Premendo RETURN senza

altra risposta, o dando una risposta inesatta, si ritorna alla prima domanda.

Il tasso di attaccare è il tempo necessario perché il suono raggiunga il suo livello più elevato. Maggiore è il valore numerico, più tempo il suono richiede per raggiungere il livello massimo. Decadere è il tempo impiegato dal suono per scendere al livello del volume di sostenere. Sostenere è il livello di volume a cui il suono rimane fino a che non inizia la sezione rilasciare. Il tasso di rilasciare è il tempo necessario al suono per scendere dal livello di sostenere fino allo zero (silenzio). Vedi figura.

Dopo che avete risposto a queste domande il calcolatore chiede il tipo di forma d'onda desiderato. Dovete battere l'iniziale del nome che contraddistingue il tipo di forma d'onda desiderato. Se si sceglie la forma d'onda vibrato (pulse), è necessario specificare la frequenza delle vibrazioni. Gli autori del manuale del Commodore 64 hanno previsto un valore di frequenza espresso da due numeri, la parte inferiore del valore di frequenza di vibrazione e la parte superiore. Per ottenere un unico valore numerico per la frequenza di vibrazione bisogna sommare alla parte inferiore del valore di frequenza la parte superiore moltiplicata per 256. Una volta che avete risposto a queste domande il calcolatore ritorna al formato normale con la voce predisposta alla forma d'onda che avete appena terminato di modificare.

La struttura del programma

Il meccanismo di programmazione è piuttosto semplice, dal momento che la maggior parte del programma è scritta in BASIC. Le istruzioni di commento REM identificano le sezioni principali del

programma (vedi la tavola per una descrizione delle variabili). Tuttavia vengono utilizzati alcuni artifici di programmazione. Il comando POKE 214,X sposta il cursore alla riga X dello schermo. Ma una istruzione PRINT senza alcun argomento deve seguire questa POKE od il cursore non si sposterà alla nuova locazione. Una istruzione POKE 788,52 disabilita il tasto RUN/STOP, ma questo fatto può risultare piuttosto noioso quando si lista il programma. Per riabilitare il tasto RUN/STOP usate il comando POKE 788,49. Anche l'istruzione WAIT viene utilizzata quando si attende una risposta (WAIT 198,255).

Il comando SYSS1 attiva (in 49152) un sottoprogramma di scansione dell'intera tastiera del Commodore 64. Questo sottoprogramma è molto utile, perché permette all'utente di premere più di un tasto per volta.

Il sottoprogramma in linguaggio macchina fornisce il codice ASCII dei tasti premuti alle locazioni 830, 831 e 832 (a causa di problemi hardware dovuti ai collegamenti dei tasti, alcune combinazioni di tasti producono valori errati). Il numero dei tasti premuti è contenuto nella locazione di memoria 829. Questo sottoprogramma può essere utilizzato in quei giochi che richiedono un ingresso multiplo.

Un secondo sottoprogramma in linguaggio macchina carica semplicemente i valori dalle locazioni da 900 a 906 nella voce opportuna del chip sonoro. Segliete il valore incrementale per le voci 0, 1 e 2 (0, 7 o 14) ed inseritelo in memoria con una POKE alla locazione 251, quindi battete SYS(49408). Il sottoprogramma non dà il via alla nota, ma lascia questo compito al BASIC, tramite una POKE al chip sonoro (SID), per la voce corrispondente.

Variabili

A contenuto vario.
A$ contenuto vario.
AD attaccare/decadere per la routine di definizione della forma d'onda.
AD() matrice dei valori di attaccare/decadere.
BF puntatore (costante) all'area tampone (198).
C$() matrice dei nomi degli accordi.
C() matrice degli scarti delle note negli accordi.
C1 numero dell'accordo.

C2 trasposizione dell'accordo.
CH segnale del formato Accordo.
ER segnale di errore del sottoprogramma di ingresso.
ET puntatore (costante) al sottoprogramma di ingresso a più tasti.
FF costante 255.
FH() matrice dei byte più significativi dei valori di frequenza.
FL() matrice dei byte meno significativi dei valori di frequenza.
HB costante 256.
I contenuto vario.
IK costante per «tasto in ingresso» o valore della matrice della tastiera.
IN valore d'ingresso dal sottoprogramma omonimo.
IN$ stringa alfanumerica d'ingresso dal sottoprogramma omonimo.
J contenuto vario.
K() matrice di conversione dei codici ASCII.
LL segnale del formato polifonico.
LN$ costante di riga.
MN segnale del formato Multivoce.
NH costante del byte più significativo alla locazione 901.
NL costante del byte meno significativo alla locazione 900.
NM$() «base», «prima» o «seconda» (per mostrare la trasposizione degli accordi).
OC numero degli scarti di mezzo passo (ottave).
P segnale del formato Mantenere.
PH() matrice dei valori più significativi della frequenza di vibrazione.
PL() matrice dei valori meno significativi della frequenza di vibrazione.
PU tasso di vibrazione per il sottoprogramma di definizione della forma d'onda.
R valore numerico della frequenza ed altri usi.
RA puntatore all'inizio del registro del formato Melodia.
RB puntatore alla fine del registro del formato Melodia.
S costante 54272.
S1 costante 49152 (per il sottoprogramma di acquisizione a più tasti).
S2 costante 49403 (per il sottoprogramma di caricamento dei

motivi).
SL segnale del formato Melodia.
SP$ costante di 39 spazi (per la cancellatura).
SR sostenere/rilasciare per il sottoprogramma di definizione della forma d'onda.
SR() matrice dei valori di sostenere/rilasciare.
T indirizzo di base attuale del chip SID.
T() matrice delle locazioni di base utilizzate per ultime.
V numero della voce del calcolatore.
VL volume.
VN costante della locazione corrispondente al numero della voce per il caricatore del motivo (251).
WF contenitore della forma d'onda per il sottoprogramma di definizione della stessa.
WV forma d'onda attuale.
WV() matrice dei valori di forma d'onda.

Tutte la variabili che iniziano con «Z» sono costanti numeriche di minore importanza.

Concertista

```
1000 GOTO1260
1010 :
1020 :
1030 REM SOTTOPROGRAMMA MELODICO
1040 IFRA<0THENRA=R
1050 RB=R:T=S+V*Z7:POKEVN,V*Z7:POKENL,FL(R
     A):POKENH,FH(RA):SYSS2:POKET+Z4,WV+Z1

1060 FORI=RATORBSTEPSGN(RB-RA)/2:POKET,FL(
     I):POKET+1,FH(I):NEXT
1070 IFPEEK(IK)=JANDPEEK(IK)-64THEN1070
1080 RA=RB:POKET+Z4,WV+P:V=V+MN*(Z1+Z3*(V=
     Z2)):RETURN
1090 :
1100 REM SOTTOPROGRAMMA ACCORDO
```

```
1110 POKEBF,Z0:FORI=Z0TOZ2:A=R+C(C1,C2,I):
     POKEVN,I*Z7:POKENL,FL(A)
1120 POKENH,FH(A):SYSS2:NEXT:POKES+Z4,WV+Z
     1:POKES+11,WV+Z1:POKES+18,WV+Z1
1130 IFPEEK(IK)=JANDPEEK(IK)-64THEN1130
1140 POKES+Z4,WV+P:POKES+11,WV+P:POKES+18,
     WV+P:RETURN
1150 :
1160 REM SOTTOPROGRAMMA POLIFONICO
1170 A=PEEK(IK):SYSS1:J=PEEK(ET):IFJ=Z0ORA
     =ZSTHENRETURN
1180 FORI=Z1TOJ:R=K(PEEK(ET+I))+OC:IFR=OCT
     HENNEXT:RETURN
1190 T(I)=V*Z7:POKEVN,T(I):POKENL,FL(R):PO
     KENH,FH(R):SYSS2
1200 IFMNTHENV=V+Z1:IFV=Z3THENV=Z0
1210 NEXT:FORI=Z1TOJ:POKES+T(I)+Z4,WV+Z1:N
     EXT
1220 SYSS1:IFJ=PEEK(ET)ANDA=PEEK(IK)THEN12
     20
1230 FORI=Z1TOJ:POKES+T(I)+Z4,WV+P:NEXT:GO
     TO1170
1240 :
1250 :
1260 REM INIZIALIZZAZIONE VARIABILI
1270 PRINT"{CLR}{CLR}{H}";:POKE53280,0:POK
     E53281,0:POKE788,52:REM IGNORA RUN/ST
     OP
1280 FORI=1TO39:SP$=SP$+" ":LN$=LN$+"[<T>]
     ":NEXT
1290 PRINT"[<5>]OTTAVA=5{ 3 SPAZI}VOCE=1 :
     A:M:M:{RVS}V{OFF}:{RVS}P{OFF}: VOLUME
     =10{DES}"LN$
1300 POKE214,23:PRINT:PRINT"DIRETTORE D'OR
     CHESTRA{ 2 SPAZI}[<8>]METCALF/SUGIYAM
     A[<5>]{HOME}{ 2 GIU'}
1310 A$="ATTENDERE PREGO[<5>]":POKE214,21:
     PRINT:PRINTTAB(12)"{WHT}"A$:S=54272:G
     OSUB2380
```

```
1320 DIMFL(134),FH(134),K(255),C(8,2,2):OC
     =48:VL=10:MN=1:LL=1:RA=-1
1330 Z1=1:Z2=2:Z3=3:Z4=4:Z7=7:ZS=64:FF=255
     :HB=256
1340 IK=197:BF=198:VN=251:NL=900:NH=901:ET
     =829:S1=49152:S2=49408:FORI=Z1TO41
1350 K(ASC(MID$("Q2W3ER5T6Y7UI9O0P@-*£↑
     {HOME}{C}ZSXDCVGBHNJM,L.:/",I)))=I:NE
     XT
1360 PRINTTAB(12)"[<8>]{SU}"A$:R=5.8:A=107
     87.4138:J=Z2↑(-Z1/12)
1370 FORI=94TO0STEP-1:FH(I)=INT(A*R/HB):FL
     (I)=A*R-HB*FH(I):A=A*J:NEXT
1380 PRINTTAB(12)"{SU}"A$:GOSUB2110
1390 :
1400 REM LETTURA DI TUTTI I DATI
1410 FORI=Z0TO8:FORJ=Z0TOZ2:READC(I,J,0),C
     (I,J,1),C(I,J,2):NEXT:READC$(I):NEXT
1420 READNM$(0),NM$(1),NM$(2):FORI=1TO8:RE
     ADAD(I),SR(I),WV(I),PL(I),PH(I):NEXT
1430 FORR=1TO2:READI,J:FORA=ITOJ:READIN:PO
     KEA,IN:NEXT:NEXT
1440 PRINTTAB(6)"{GIU'}(USATE CONTROL-X PE
     R USCIRE)":I=1:GOSUB1660
1450 :
1460 :
1470 REM NUCLEO
1480 WAITBF,FF:J=PEEK(IK):GETA$:R=K(ASC(A$
     ))+OC:IFR=OCTHENGOSUB1600:GOTO1480
1490 IFSLTHENGOSUB1040:GOTO1480
1500 IFCHTHENGOSUB1110:GOTO1480
1510 IFLLTHENGOSUB1170:GOTO1480
1520 T=S+V*Z7:POKEVN,V*Z7:POKENL,FL(R):POK
     ENH,FH(R):SYSS2:POKET+Z4,WV+Z1
1530 IFMNTHENV=V+Z1:IFV=Z3THENV=Z0
1540 IFPEEK(IK)=JANDPEEK(IK)-64THEN1540
1550 POKET+Z4,WV+P:WAITBF,FF:GETA$:J=PEEK(
     IK):R=K(ASC(A$))+OC:IFR-OCTHEN1520
1560 GOSUB1600:GOTO1480
```

```
1570 :
1580 :
1590 REM PARAMETRI DELLE FUNZIONI
1600 IFCH=0THEN1630
1610 FORI=0TO2:IFA$=MID$("+-£",I+1,1)THENC
     2=I:PRINT"{HOME}{GIU'}"TAB(32)NM$(I):
     RETURN
1620 NEXT:A=ASC(A$):IFA>32ANDA<42THENC1=A-
     33:PRINT"{HOME}{GIU'}"TAB(8)C$(C1):RE
     TURN
1630 FORI=1TO8:IFA$<>MID$("{BLK}{WHT}{RED}
     {CYN}{PUR}{GRN}{BLU}{YEL}",I,1)THENNE
     XT:GOTO1650
1640 OC=12*(I-Z1):PRINT"{HOME}"TAB(7)MID$(
     STR$(I),2):RETURN
1650 FORI=1TO8:IFA$<>MID$("[<1>][<2>][<3>]
     [<4>][<5>][<6>][<7>][<8>]",I,1)THENNE
     XT:GOTO1680
1660 POKE902,PL(I):POKE903,PH(I):WV=WV(I):
     POKE904,WV:POKE905,AD(I):POKE906,SR(I
     )
1670 PRINT"{HOME}"TAB(16)MID$(STR$(I),2):R
     ETURN
1680 IFA$<>"{F1}"ANDA$<>"{F3}"THEN1730
1690 VL=VL-(VL<15ANDA$="{F1}")+(VL>0ANDA$=
     "{F3}"):POKES+24,VL
1700 PRINT"{HOME}"TAB(37)RIGHT$("0"+MID$(S
     TR$(VL),2),2):RETURN
1710 :
1720 REM FUNZIONI DI ARRANGIAMENTO
1730 IFA$="{F4}"THENP=1-P:POKE1047,13+128*
     P:GOTO2380
1740 IFA$="{F6}"THENMN=1-MN:POKE1049,22+12
     8*MN:GOTO2380
1750 IFA$="{F8}"THENLL=1-LL:POKE1051,16+12
     8*LL:RETURN
1760 IFA$="{F7}"THENSL=1-SL:RA=-1:POKE1045
     ,13+128*SL:CH=1:GOTO1790
1770 IFA$<>"{F5}"THEN1810
```

```
1780 POKE1045,13:SL=0
1790 CH=1-CH:POKE1043,1+128*CH:IFCH=0THENP
     RINT"{HOME}{GIU'}"LN$:PRINTSP$:RETURN

1800 PRINT"{HOME}{GIU'}"SP$"{DES}{SU}ACCOR
     DO:"C$(C1);
1805 PRINTTAB(21)"TRASPOSIZ.:"NM$(C2)"
     {DES}"LN$:RETURN
1810 IFA$=" "THENGOSUB2380:RA=-1:POKEBF,Z0
     :RETURN
1820 IFA$="{X}"THENGOSUB2380:PRINT"{CLR}";
     :POKE788,49:END
1830 IFA$<>"{F2}"THENRETURN
1840 :
1850 :
1860 REM VISUALIZZAZIONE DEI PARAMETRI
             DELLA FORMA D'ONDA
1870 GOSUB2270:POKE214,13:PRINT
1880 PRINT"VOCE DA DEFINIRE (1-8)";:J=1:GO
     SUB2300
1890 IFIN<1ORIN>8THENGOSUB2270:GOTO2200
1900 I=IN:PRINTTAB(31)"ATT:"MID$(STR$(INT(
     AD(I)/16)),2)
1910 PRINTTAB(31)"DEC:"MID$(STR$(AD(I)AND1
     5),2)
1920 PRINTTAB(31)"SOS:"MID$(STR$(INT(SR(I)
     /16)),2)
1930 PRINTTAB(31)"RIL:"MID$(STR$(SR(I)AND1
     5),2)
1940 PRINTTAB(31)"FDO:[<8>]"MID$("DENTRIVI
     BRUM",3*LOG(WV(I))/LOG(2)-11,3)"[<5>]

1950 IFWV(I)=64THENPRINTTAB(31)"VIB:"MID$(
     STR$(PH(I)*HB+PL(I)),2)
1960 :
1970 REM DEFINISCE UNA NUOVA
             FORMA D'ONDA
1980 POKE214,14:PRINT:PRINT"TASSO DI ATTAC
     CARE (0-15)";:J=2
```

```
1985 GOSUB2300:IFERTHEN1870
1990 AD=IN:PRINT"TASSO DI DECADERE (0-15)"
     ;:GOSUB2300:IFERTHEN1870
2000 AD=AD*16ORIN:PRINT"LIV. DI SOSTENERE
     (0-15)";:GOSUB2300:IFERTHEN1870
2010 SR=IN:PRINT"TASSO DI RILASCIARE (0-15
     )";:GOSUB2300:IFERTHEN1870
2020 SR=SR*16ORIN:PRINT"[<8>]D[<5>]ENTE
     [<8>]T[<5>]RIANG. [<8>]V[<5>]IBR.
     [<8>]R[<5>]UM.";:J=1:GOSUB2300
2030 FORJ=1TO4:IFIN$<>MID$("DTVR",J,1)THEN
     NEXT:GOTO1870
2040 WF=2↑(J+3):IFWF<>64THEN2060
2050 PRINT"TASSO DI VIBR. (0-4095)";:J=4:G
     OSUB2300
2055 PU=IN:IFIN<0ORIN>4095THEN1870
2060 WV(I)=WF:PL(I)=PU-HB*INT(PU/HB):PH(I)
     =INT(PU/HB):AD(I)=AD:SR(I)=SR
2070 GOSUB2270:GOSUB2210:GOTO1660
2080 :
2090 :
2100 REM VISUALIZZAZIONE DELLA TASTIERA
2110 POKES+24,VL:PRINT"{HOME}{ 3 GIU'}"TAB
     (9)"[<M>]{RVS} {DES} {DES} _ {DES}
     {DES} {DES} _ {DES} {DES} _ {DES}
     {DES} {DES} "
2120 PRINT"TASTIERA [<M>]{RVS} {OFF}2{RVS}
      {OFF}3{RVS} _ {OFF}5{RVS} {OFF}6
     {RVS} {OFF}7{RVS} _ {OFF}9{RVS} {OFF}
     0{RVS} _ {OFF}-{RVS} {OFF}£{RVS} S "
2130 PRINT"SUPERIORE[<M>]{RVS}{SPAZI}_
     {SPAZI}_{SPAZI}_{SPAZI}_{SPAZI}_
     {SPAZI}_{SPAZI}_{SPAZI}_{SPAZI}_
     {SPAZI}_{SPAZI}_{SPAZI}_ _ "
2140 PRINTTAB(9)"[<M>]{RVS}Q_W_E_R_T_Y_U_
     I_O_P_@_*_↑_ "
2150 PRINTTAB(13)"{GIU'}[<N>]{RVS} {DES}
     {DES} _ {DES} {DES} {DES} _ {DES}
     {DES} {OFF}[<H>]"
```

```
2160 PRINT"TASTIERA{ 5 SPAZI}[<N>]{RVS}
     {OFF}S{RVS} {OFF}D{RVS} _ {OFF}G{RVS}
      {OFF}H{RVS} {OFF}J{RVS} _ {OFF}L
     {RVS} {OFF}:{RVS} {OFF}[<H>]"
2170 PRINT"INFERIORE{ 4 SPAZI}[<M>]{RVS}
     _ _ _ _ _{SPAZI}_ _ _ _ {OFF}[<H>]"
2180 PRINTTAB(13)"[<N>]{RVS}Z_X_C_V_B_N_M
     _,_._/{OFF}[<H>]"
2190 :
2200 REM VISUALIZZAZIONE DELLE FUNZIONI
              DI MENU
2210 POKE214,13:PRINT:PRINT"F1 -- VOLUME +
     { 2 SPAZI}F2 -- FORMA D'ONDA"
2220 PRINT"{GIU'}F3 -- VOLUME -{ 2 SPAZI}
     F4 -- [<8>]MANTENERE[<5>]"
2230 PRINT"{GIU'}F5 -- [<8>]ACCORDI[<5>]
     { 3 SPAZI}F6 -- [<8>]MULTIVOCE[<5>]"
2240 PRINT"{GIU'}F7 -- [<8>]MELODIE[<5>]
     { 3 SPAZI}F8 -- [<8>]POLIFONIA[<5>]":
     RETURN
2250 :
2260 REM CANCELLA L'AREA
              DI VISUALIZZAZIONE
2270 POKE214,12:PRINT:FORJ=1TO11:PRINTSP$:
     NEXT:RETURN
2280 :
2290 REM SOTTOPROGRAMMA D'INGRESSO
2300 IN$="":PRINT"? ";
2310 PRINT"{RVS} {OFF}{SIN}";:WAITBF,FF:GE
     TA$:IFA$="{X}"THEN1820
2320 A=ASC(A$):IFA=13THENPRINT" ":IN=VAL(I
     N$):ER=(IN<0ORIN>15)ORIN$="":RETURN
2330 IFA=20ANDLEN(IN$)THENPRINT" { 2 SIN}
      {SIN}";:IN$=LEFT$(IN$,LEN(IN$)-1)
2340 IF(AAND127)<35ORLEN(IN$)=JTHEN2310
2350 PRINTA$;:IN$=IN$+A$:GOTO2310
2360 :
2370 REM CLEAR MUSIC CHIP
2380 FORI=4TO18STEP7:POKES+I,0:NEXT:FORI=0
```

```
     TO23:POKES+I,0:NEXT:RETURN
2390 :
2400 :
2410 REM DATI DEGLI ACCORDI
2420 DATA,4,7,,3,8,,5,9,"MAGGIORE
     { 2 SPAZI}",,3,7,,4,9,,5,8,"MINORE
     { 4 SPAZI}"
2430 DATA,3,6,,3,9,,6,9,"DIMINUITO ",,4,8,
     ,4,8,,4,8,"AUMENTATO "
2440 DATA,4,11,,4,11,,4,11,"7.MAGGIORE",,3
     ,10,,3,10,,3,10,"7. MINORE "
2450 DATA,4,10,,4,10,,4,10,"7. DOMIN. ",4,
     7,9,4,7,9,4,7,9,"6.MAGGIORE"
2460 DATA3,7,9,3,7,9,3,7,9,"6. MINORE ","B
     ASE{ 3 SPAZI}","PRIMA{ 2 SPAZI}","SEC
     ONDA"
2470 :
2480 REM DATI DEI PARAMETRI
            DELLE FORME D'ONDA
2490 DATA,249,16,,,249,32,,,249,64,160,1
     5,,249,128,,,240,16,,204,204,16,,
2500 DATA,252,64,200,,192,240,32,,
2510 :
2520 REM CODICE ASSEMBLATORE
            MULTI-INGRESSO
2530 DATA49152,49294,120,169,,141,61,3,170
     ,169,254,133,252,165,252,141,,220,173

2540 DATA1,220,157,143,192,232,56,38,252,1
     76,239,162,,160,,189,143,192,42,176
2550 DATA29,72,132,253,138,10,10,10,5,253,
     168,185,79,192,238,61,3,172,61,3,153
2560 DATA61,3,104,192,3,240,12,164,253,200
     ,192,8,208,219,232,224,8,208,209,88
2570 DATA96,17,135,134,133,136,29,13,20,0,
     69,83,90,52,65,87,51,88,84,70,67,54
2580 DATA68,82,53,86,85,72,66,56,71,89,55,
     78,79,75,77,48,74,73,57,44,64,58,46
2590 DATA45,76,80,43,47,94,61,1,19,59,42,9
```

```
     2,3,81,2,32,50,4,95,49
2600 :
2610 REM CODICE ASSEMBLATORE
              LETTORE MOTIVI
2620 DATA49408,49454,169,212,133,252,169,,
     160,6,145,251,136,145,251,170,169,8
2630 DATA136,145,251,138,145,251,136,192,1
     ,208,249,188,41,193,185,132,3,145,251

2640 DATA232,224,6,208,243,96,2,3,,1,6,5
```

Appendici

Appendice A

Una guida dedicata ai principianti

Cos'è un programma?

Un calcolatore non può svolgere un compito autonomamente. Come un'automobile priva di carburante, un calcolatore ha un *potenziale*, ma senza un programma non ottiene alcun risultato. La maggior parte dei programmi pubblicati in questo libro sono scritti in un linguaggio di programmazione chiamato BASIC. Il BASIC è facile da imparare ed è incorporato nel C-64.

Programmi BASIC

I calcolatori possono essere piuttosto pignoli. Contrariamente alla lingua italiana, che è ricca di ambiguità, il BASIC generalmente prevede un solo modo esatto di formulare qualcosa. Ogni lettera, carattere o numero è significativo. Un errore piuttosto diffuso consiste nel sostituire una lettera come la O al numero 0, una l minuscola al numero 1 od una B maiuscola al numero 8. Inoltre dovete

copiare correttamente tutti i segni di interpunzione, come i due punti ":" e le virgole ",", proprio come compaiono nel libro. Le spaziature possono essere molto importanti. Per essere sicuri copiate i listati *esattamente* come appaiono.

Parentesi e caratteri speciali

L'eccezione a queste regole di copiatura è rappresentata dalle parentesi, come [GIÙ]. Tutto ciò che è racchiuso tra parentesi rappresenta un carattere speciale o caratteri che non possono

essere facilmente listati con una stampante. Quando incontrate una di queste istruzioni speciali, fate riferimento all'Appendice B «Come copiare i programmi».

Qualche nota riguardo le istruzioni DATA

Alcuni programmi contengono uno o più sezioni di istruzioni DATA. Queste righe forniscono informazioni necessarie al programma. Alcune istruzioni DATA contengono parti di programma effettive (chiamate linguaggio macchina); altre contengono codici simbolici grafici. Queste righe sono particolarmente esposte ad errori.

Se anche un solo numero in una qualsiasi delle istruzioni DATA è copiato in maniera errata, la vostra macchina può bloccarsi. La tastiera e il tasto STOP possono sembrare inerti e lo schermo può scomparire. Non fatevi prendere dal panico: non si è prodotto alcun danno. Per riacquistare il controllo dovete spegnere il calcolatore, quindi riaccenderlo. Ciò cancellerà qualsiasi programma contenuto in memoria; *quindi registrate sempre una copia dei vostri programmi prima di mandarli in esecuzione.* Se il vostro calcolatore si blocca, potete ricaricare il programma e andare in cerca degli errori che vi sono contenuti.

Talvolta una istruzione DATA copiata in maniera non corretta provocherà un messaggio di errore quando il programma viene mandato in esecuzione. Il messaggio di errore può far riferimento alla riga di programma che legge (con una istruzione READ) i dati. *L'errore, comunque, è ancora contenuto nelle istruzioni DATA.*

Imparate a conoscere a fondo il vostro calcolatore

Dovreste familiarizzare con il vostro calcolatore prima di provare a copiare un programma. Imparate tutto ciò che riguarda le istruzioni usate per registrare e recuperare i programmi da nastro o da disco. Dovrete registrare una copia dei vostri programmi, in modo da non doverli inserire nel calcolatore tutte le volte che volete usarli. Imparate ad usare le opzioni che permettono di operare le correzioni necessarie sul vostro calcolatore. Come potete correggere una riga, se avete commesso un errore? Potete sempre ribattere la riga, ma dovete perlomeno sapere come fare a tornare indietro del numero di caratteri necessari. Sapete inserire i caratteri in negativo, in minuscolo ed i caratteri di controllo? È tutto chiaramente spiegato nel manuale del vostro calcolatore.

Appendice A

Un rapido riassunto

1) Copiate il programma una riga per volta, in ordine. Premete RETURN alla fine di ciascuna riga. Usate INST/DEL o i tasti cursore per correggere gli errori.
2) Confrontate la riga che avete appena copiato con la riga di programma riportata dal libro. Potete ricontrollare l'intero programma da capo, se ottenete un messaggio di errore durante l'esecuzione del programma.

Appendice B

Come copiare i programmi

Per facilitare la comprensione di ciò che si deve battere quando copiate uno di questi programmi sul vostro calcolatore, abbiamo stabilito le seguenti convenzioni circa i listati.

Generalmente i listati dei programmi del Commodore 64 conterranno parole racchiuse tra parentesi, che segnalano ciascun carattere speciale: [GIÙ] indica premere il tasto CRSR di sinistra; [5 SPAZI] indica premere la barra spaziatrice cinque volte.

Per indicare che un tasto va *premuto insieme al tasto SHIFT* (vale a dire tenere premuto il tasto SHIFT mentre si preme l'altro tasto) il tasto in questione apparirà sottolineato nei nostri listati. Ad esempio, una S sottolineata indica premere il tasto S mentre si tiene premuto il tasto SHIFT. Ciò apparirà sullo schermo come un simbolo a forma di cuore. Se trovate un tasto sottolineato racchiuso tra parentesi (ad esempio, [10 N (sottolineato)]), dovreste premere il tasto il numero di volte indicato (nel nostro caso, dovreste battere dieci N accoppiate al tasto SHIFT).

Se un tasto è racchiuso da parentesi speciali [()], dovete tenere premuto il *tasto Commodore* mentre premete il tasto racchiuso dalle parentesi stesse (il tasto Commodore si trova nell'angolo inferiore sinistro della tastiera). Anche in questo caso, se il tasto è preceduto da un numero, dovete premerlo il numero di volte necessario.

Raramente vedrete una singola lettera dell'alfabeto posta tra parentesi. Questi caratteri possono essere inseriti tenendo premuto il tasto CTRL mentre si batte la lettera racchiusa fra le parentesi. Ad esempio, [A] indica dover premere CTRL-A.

Per quanto riguarda il *formato doppi apici,* sapete che siete in grado di muovere il cursore per lo schermo con i tasti CRSR. A volte un programmatore desidera spostare il cursore sotto controllo del programma. Ecco perché vedrete i vari [SINISTRA], [HOME] e [BLU] nei nostri programmi. L'unico modo che il calcolatore ha di distinguere tra controllo cursore diretto e controllo cursore da pro-

gramma è rappresentato dal formato «doppi apici».

Una volta premuti i doppi apici (SHIFT-2) vi trovate nel formato omonimo. Se battete qualsiasi cosa e quindi cercate di modificarla spostando il cursore verso sinistra, otterrete solo una serie di caratteri in negativo. Questi sono simboli corrispondenti a spostamenti del cursore verso sinistra. L'unico tasto di correzione non programmabile è il tasto INST/DEL; quindi potete continuare ad usare il tasto INST/DEL per tornare sui caratteri da correggere. Una volta premuti nuovamente i doppi apici uscite dal formato «doppi apici».

Vi ponete in formato «doppi apici» anche quando inserite spazi in una riga di programma tramite la coppia di tasti SHIFT-INST/DEL. In ogni caso, il modo più semplice per uscire dal formato doppi apici consiste nel premere RETURN. Vi troverete al di fuori del formato «doppi apici» e potrete spostarvi con i tasti cursore sulla riga errata e correggerla.

Usate la tavola seguente quando dovete inserire tasti controllo cursore o colore.

Quando leggete:	**Premete:**	**Vedrete:**	**Quando leggete:**	**Premete:**	**Vedrete:**
{CLR} O [CLR]	SHIFT CLR/HOME		[<1>]	C= 1	
{HOME} O [HOME]	CLR/HOME		[<2>]	C= 2	
{SU} O [SU]	SHIFT ⇑ CRSR ⇓		[<3>]	C= 3	
{GIU'} O [GIU']	⇑ CRSR ⇓		[<4>]	C= 4	
{SIN} O [SIN]	SHIFT ⇐CRSR⇒		[<5>]	C= 5	
{DES} O [DES]	⇐CRSR⇒		[<6>]	C= 6	
{RVS} O [RVS]	CTRL 9		[<7>]	C= 7	
{OFF} O [OFF]	CTRL 0		[<8>]	C= 8	
{BLK} O [BLK]	CTRL 1		{F1} O [F1]	f1	
{WHT} O [WHT]	CTRL 2		{F2} O [F2]	f2	
{RED} O [RED]	CTRL 3		{F3} O [F3]	f3	
{CYN} O [CYN]	CTRL 4		{F4} O [F4]	f4	
{PUR} O [PUR]	CTRL 5		{F5} O [F5]	f5	
{GRN} O [GRN]	CTRL 6		{F6} O [F6]	f6	
{BLU} O [BLU]	CTRL 7		{F7} O [F7]	f7	
{YEL} O [YEL]	CTRL 8		{F8} O [F8]	f8	

Appendice C

Tavola delle locazioni di memoria dello schermo

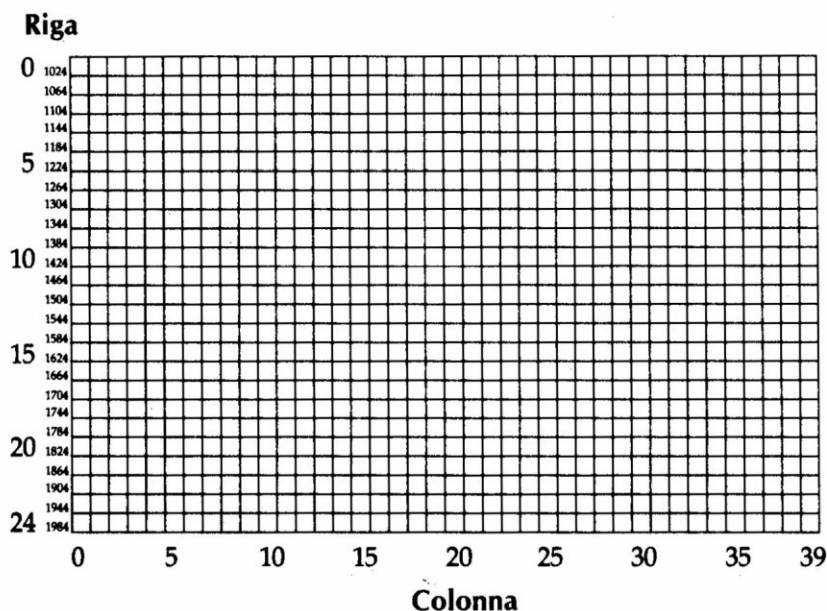



Tavola delle locazioni di memoria del colore di schermo

Appendice E

Tavola dei valori che identificano i colori

Valori da utilizzare nelle istruzioni POKE per ciascun colore

Colore	Nybble infer. (4 bit) codice colore	Nybble super. (4 bit) codice colore	Valore di selezione colore in formato multicolore
Nero	0	0	8
Bianco	1	16	9
Rosso	2	32	10
Blu-verde	3	48	11
Porpora	4	64	12
Verde	5	80	13
Blu	6	96	14
Giallo	7	112	15
Arancio	8	128	—
Marrone	9	144	—
Rosso chiaro	10	160	—
Grigio scuro	11	176	—
Grigio medio	12	192	—
Verde chiaro	13	208	—
Azzurro	14	224	—
Grigio chiaro	15	240	—

Appendice E

Locazioni in cui inserire i codici colore tramite istruzioni POKE per ciascun formato.

Formato*	**Bit o coppia di Bit**	**Locazione**	**Codice colore**
Testo normale	0	53281	4 inf.
	1	Memoria colore	4 inf.
Testo multicolore	00	53281	4 inf.
	01	53282	4 inf.
	10	53283	4 inf.
	11	Memoria colore	Scelta colore
Colore di fondo esteso #	00	53281	4 inf.
	01	53282	4 inf.
	10	53283	4 inf.
	11	53284	4 inf.
Mappa di bit	0	Memoria schermo	± 4 inf.
	1	Memoria schermo	± 4 sup.
Mappa di bit multicolore	00	53281	4 inf.
	01	Memoria schermo	± 4 sup.
	10	Memoria schermo	± 4 inf.
	11	Memoria colore	4 inf.

Con il termine «4 inf.» si intendono i 4 bit meno significativi (inferiori); con «4 sup.» i 4 bit più significativi (superiori).

* Per tutti i formati il colore della cornice dello schermo è controllato inserendo il codice colore, rappresentato dai quattro bit meno significativi, nella locazione 53280, tramite una istruzione POKE.

In formato «colore di fondo esteso» i bit 6 e 7 di ciascun byte della memoria di schermo funzionano come coppia di bit che controlla il colore di fondo. Dato che solo i bit da 0 a 5 sono disponibili per la scelta del carattere, solo i caratteri con un codice simbolico di schermo compreso tra 0 e 63 possono essere utilizzati in questo formato.

$\pm$ In formato a mappa di bit viene applicato l'operatore OR ai quattro bit più significativi ed ai quattro bit meno significativi ed il risultato viene inserito, tramite una istruzione POKE, nella stessa locazione della memoria di schermo, per controllare il colore della cella corrispondente della mappa di bit. Ad esempio, per controllare i colori della cella 0 della mappa di bit applicate l'operatore OR ai valori numerici dei quattro bit più significativi e dei quattro bit meno significativi ed inserite il risultato con una POKE nella locazione 0 di memoria di schermo.

Codici ASCII

ASCII	CARATTERE	ASCII	CARATTERE
5	BIANCO	50	2
8	DISABILITA	51	3
	SHIFT COMMODORE	52	4
9	ABILITÀ	53	5
	SHIFT COMMODORE	54	6
13	RETURN	55	7
14	MINUSCOLO	56	8
17	CURSORE VERSO IL BASSO	57	9
18	NEGATIVO	58	:
19	HOME	59	;
20	DELETE	60	<
28	ROSSO	61	=
29	CURSORE VERSO DESTRA	62	>
30	VERDE	63	?
31	BLU	64	@
32	SPAZIO	65	A
33	!	66	B
34	"	67	C
35	#	68	D
36	$	69	E
37	%	70	F
38	&	71	G
39	'	72	H
40	(	73	I
41	)	74	J
42	*	75	K
43	+	76	L
44	,	77	M
45	–	78	N
46	.	79	O
47	/	80	P
48	0	81	Q
49	1	82	R

Appendice F

ASCII	CARATTERE	ASCII	CARATTERE
83	S	118	
84	T	119	
85	U	120	♣
86	V	121	
87	W	122	♦
88	X	123	
89	Y	124	
90	Z	125	
91	[	126	π
92	£	127	
93	]	129	ARANCIO
94	↑	133	f1
95	←	134	f3
96		135	f5
97	♠	136	f7
98		137	f2
99		138	f4
100		139	f6
101		140	f8
102		141	SHIFT + RETURN
103		142	MAIUSCOLO
104		144	NERO
105		145	CURSORE VERSO L'ALTO
106		146	DISINSERISCE IL NEGATIVO
107		147	CANCELLA LO SCHERMO
108		148	INSERIMENTO
109		149	MARRONE
110		150	ROSSO CHIARO
111		151	GRIGIO SCURO
112		152	GRIGIO MEDIO
113	●	153	VERDE CHIARO
114		154	AZZURRO
115	♥	155	GRIGIO CHIARO
116		156	PORPORA
117		157	CURSORE VERSO SINISTRA

Appendice F

ASCII	CARATTERE
158	GIALLO
159	BLU-VERDE
160	SPAZIO
161	
162	
163	
164	
165	
166	
167	
168	
169	
170	
171	
172	
173	
174	
175	
176	
177	
178	
179	
180	
181	
182	
183	
184	
185	
186	
187	
188	
189	
190	
191	
192	
193	
194	
195	
196	
197	
198	
199	
200	
201	
202	
203	
204	
205	
206	
207	
208	
209	
210	
211	
212	
213	
214	
215	
216	
217	
218	
219	
220	
221	
222	
223	
224	SPAZIO
225	
226	
227	

Appendice F

ASCII	CARATTERE
228	
229	
230	
231	
232	
233	
234	
235	
236	
237	
238	
239	
240	
241	
242	
243	
244	
245	
246	
247	
248	
249	
250	
251	
252	
253	
254	
255	π

I valori di codice 0-4, 6, 7, 10-12, 15, 16, 21-27, 128, 130-132 e 143 sono inutilizzati.

Codici di schermo

POKE	Maiuscole e set grafico completo	Maiuscole e minuscole
0	@	@
1	A	a
2	B	b
3	C	c
4	D	d
5	E	e
6	F	f
7	G	g
8	H	h
9	I	i
10	J	j
11	K	k
12	L	l
13	M	m
14	N	n
15	O	o
16	P	p
17	Q	q
18	R	r
19	S	s
20	T	t
21	U	u
22	V	v
23	W	w
24	X	x
25	Y	y
26	Z	z
27	[	[
28		
29	]	]
30	↑	↑

Appendice G

POKE	Maiuscole e set grafico completo	Maiuscole e minuscole
31	←	←
32	-spazio-	
33	!	!
34	"	"
35	#	#
36	$	$
37	%	%
38	&	&
39	'	'
40	(	(
41	)	)
42	*	*
43	+	+
44	,	,
45	–	–
46	.	.
47	/	/
48	0	0
49	1	1
50	2	2
51	3	3
52	4	4
53	5	5
54	6	6
55	7	7
56	8	8
57	9	9
58	:	:
59	;	;
60	<	<
61	=	=
62	>	>
63	?	?
64		

POKE	Maiuscole e set grafico completo	Maiuscole e minuscole
65	♠	A
66		B
67		C
68		D
69		E
70		F
71		G
72		H
73		I
74		J
75		K
76		L
77		M
78		N
79		O
80		P
81	●	Q
82		R
83	♥	S
84		T
85		U
86		V
87	○	W
88	♣	X
89		Y
90	♦	Z
91		
92		
93		
94	π	
95		
96	-spazio-	
97		
98		

Appendice G

POKE	Maiuscole e set grafico completo	Maiuscole e minuscole
99		
100		
101		
102		
103		
104		
105		
106		
107		
108		
109		
110		
111		
112		
113		

POKE	Maiuscole e set grafico completo	Maiuscole e minuscole
114		
115		
116		
117		
118		
119		
120		
121		
122		
123		
124		
125		
126		
127		

I caratteri aventi numeri di codice da 128 a 255 sono l'immagine in negativo dei caratteri aventi numeri di codice da 0 a 127.

Codici dei tasti del Commodore 64

Commodore 64 Keycodes

Key	Keycode	Key	Keycode
A	10	L	42
B	28	M	36
C	20	N	39
D	18	O	38
E	14	P	41
F	21	Q	62
G	26	R	17
H	29	S	13
I	33	T	22
J	34	U	30
K	37	V	31

Appendice H

Key	Keycode	Key	Keycode
W	9	@	46
X	23	*	49
Y	25	↑	54
Z	12	:	45
1	56	;	50
2	59	=	53
3	8	RETURN	1
4	11	,	47
5	16	.	44
6	19	/	55
7	24	CRSR ↑↓	7
8	27	CRSR ⇄	2
9	32	f1	4
0	35	f3	5
+	40	f5	6
–	43	f7	3
£	48	SPAZIO	60
CLR/HOME	51	RUN/STOP	63
INST/DEL	0	NESSUN TASTO PREMUTO	64
←	57		

Il codice dei tasti è il valore numerico contenuto nella locazione di memoria 197 e rappresenta il tasto attualmente premuto. Provate questo programma di una sola riga:

```
10 PRINT PEEK(197): GOTO 10
```

Come usare MLX: un programma per copiare programmi in linguaggio macchina

Charles Brannon

Vi ricordate dell'ultima volta in cui avete copiato un programma in linguaggio macchina? Avete battuto centinaia di istruzioni DATA, numeri e virgole. Malgrado la fatica fatta, non potevate essere sicuri di averlo copiato esattamente. Quindi siete ritornati da capo a ricontrollare il programma, l'avete mandato in esecuzione, siete rimasti bloccati, siete tornati a ricontrollarlo da capo un'altra volta, avete corretto qualche errore di battitura, l'avete nuovamente mandato in esecuzione, siete rimasti di nuovo bloccati, avete ricontrollato ciò che avevate battuto... Piuttosto frustrante, non è vero?

Finora, tuttavia, questo è stato il sistema migliore per copiare programmi in linguaggio macchina nel vostro calcolatore. A meno che possediate un assemblatore ed abbiate voglia di cimentarvi con la programmazione in linguaggio macchina a livello assemblaggio, è molto più semplice copiare un programma BASIC che legga delle frasi DATA e ne inserisca i valori numerici in memoria con delle istruzioni POKE.

Appendice I

Alcuni di questi «Caricatori BASIC» usano una somma di controllo per verificare che abbiate copiato i valori numerici correttamente. La forma più semplice di somma di controllo consiste nel sommare semplicemente tutti i valori numerici contenuti nelle istruzioni DATA. Se commettete un errore, la somma di controllo non coinciderà. Alcuni programmatori hanno facilitato il vostro compito creando somme di controllo ogni dieci righe, in modo da poter identificare esattamente i vostri eventuali errori.

Non è finita! MLX vi viene in aiuto! MLX è il modo migliore di copiare tutti quei lunghi programmi in linguaggio macchina con un minimo sforzo. MLX vi permette di copiare i valori numerici riportati da un listato speciale che ha l'aspetto di istruzioni DATA BASIC. Controlla le vostre battute riga per riga. Non vi lascia battere caratteri non ammessi quando dovreste battere numeri. Non vi lascia inserire numeri maggiori di 255. Vi impedisce di copiare un numero sbagliato da una riga sbagliata. In poche parole, MLX rende superfluo ricontrollare ciò che si è battuto.

L'uso dell'unità dischi

In più, MLX produrrà un file su nastro o su disco, pronto per l'uso. Potete quindi usare il comando LOAD per leggere il programma all'interno della memoria del calcolatore, proprio come un qualsiasi programma. Per essere più precisi, dovete battere:

LOAD «nome del programma»,1,1

(per l'unità nastro)

o

LOAD «nome del programma»,8,1

(per l'unità dischi)

Per mandare in esecuzione il programma, dovete battere un comando SYS che trasferisca il controllo dal BASIC al linguaggio macchina. L'indirizzo di partenza segnalato dalla SYS verrà sempre citato nell'articolo in cui appare il programma in linguaggio macchina.

Come si usa MLX

Copiate e registrate MLX (vi tornerà utile in futuro). Quando siete pronti per copiare un programma in linguaggio macchina mandate MLX in esecuzione. Il programma vi chiederà due valori numerici: l'indirizzo iniziale e quello conclusivo. Questi numeri sono descritti

nel testo relativo ai vari programmi.

Il programma indicherà la riga in cui state attualmente copiando i valori numerici ricavabili dal listato. Ogni riga è composta da sei numeri più una somma di controllo. Se copiate in maniera errata uno qualsiasi dei sei numeri, oppure la somma di controllo, il C64 farà suonare un cicalino e vi chiederà di ribattere la riga. Se la copiate correttamente, suonerà un campanello e potrete continuare, copiando la riga successiva.

Un programma speciale per la correzione degli errori

Con MLX non viene usato il normale programma per la correzione degli errori del Commodore 64. Ad esempio, come ingresso verranno accettati solo numeri. Se dovete operare una correzione, premete il tasto INST/DEL; verrà cancellato il numero completo. Potete premere questo tasto il numero di volte necessario, fino a tornare all'inizio della riga. Se battete numeri di tre cifre, come riportato dai listati, il calcolatore inserirà automaticamente le virgole e proseguirà acquisendo il numero successivo. Se battete meno di tre cifre, potete premere sia la virgola che la barra spaziatrice od il tasto RETURN per avanzare fino al numero successivo. La somma di controllo apparirà automaticamente in negativo; non preoccupatevene: viene evidenziata appositamente.

Quando ho provato MLX ho trovato estremamente facile copiare lunghi listati. Con l'ausilio dei segnali acustici forniti non dovreste nemmeno aver bisogno di guardare lo schermo, se siete dei buoni dattilografi.

Ce l'ho fatta, finalmente!

Quando avete terminato il vostro lavoro di copiatura, presumendo che lo facciate in una sola volta, potete registrare su nastro o disco il programma completato ed esente da errori. Seguite le istruzioni che compaiono sullo schermo. Se nel corso della regi strazione ottenete delle segnalazioni di errore, avete probabilmente utilizzato un disco in cattive condizioni, oppure il disco non ha più spazio disponibile, oppure avete commesso un errore nel copiare lo stesso programma MLX (ci dispiace, ma non è in grado di controllare se stesso!).

Appendice I

Comandi di controllo

Che cosa dovete fare, se non volete copiare tutto il programma in una sola volta? MLX vi consente di copiarne quanto volete, registrare ciò che avete fatto e quindi ricaricare il file da nastro o da disco quando volete continuare. MLX accetta solo questi comandi:

SHIFT-S: Registrazione (SAVE)
SHIFT-L: Lettura (LOAD)
SHIFT-N: Nuovo indirizzo
SHIFT-D: Visualizzazione

Tenete premuto il tasto SHIFT mentre premete il tasto opportuno. Uscirete dalla riga che state copiando, quindi vi consiglierei di farlo all'inizio di una nuova riga. Usate il comando SHIFT-S per registrare ciò che avete copiato. Ciò realizzerà un file su nastro o su disco come se aveste finito, ma il programma contenuto nel file, ovviamente, non funzionerà fino a che avrete terminato di copiarlo. Ricordatevi a quale indirizzo vi siete interrotti. Quando, in seguito, riutilizzerete MLX rispondete a tutte le domande come avete fatto in precedenza, quindi caricate il file da nastro o da disco. Quando vi viene proposta la riga iniziale premete SHIFT-L per caricare il file in memoria. Utilizzerete quindi il comando «Nuovo indirizzo» per riprendere a copiare dal punto in cui vi eravate interrotti.

Nuovo indirizzo e visualizzazione

Dopo aver premuto SHIFT-N battete l'indirizzo a cui vi siete precedentemente arrestati. La riga mostrata cambierà e voi potrete quindi continuare a copiare. Inserite sempre un nuovo indirizzo che corrisponda ad uno dei numeri di riga del listato, altrimenti la somma di controllo non coinciderà. Potete usare il comando di visualizzazione per far apparire una parte del vostro lavoro. Dopo aver premuto SHIFT-D indicate due indirizzi all'interno del campo di valori del listato per quanto riguarda il numero di riga. Potete interrompere il listato premendo un tasto qualsiasi.

Qualche possibile inconveniente

Questi comandi speciali possono sembrare un po' complicati, ma, man mano che familiarizzerete con MLX, diventeranno pre-

ziosi. Ad esempio, come fare se vi siete dimenticati in che punto vi siete interrotti? Usate il comando di visualizzazione per scandire la memoria dall'inizio del programma fino alla fine. Quando trovate una serie di 170, interrompete il listato (premendo un tasto) e proseguite il vostro lavoro di copiatura a partire dal punto in cui iniziano i 170. Alcuni programmi contengono delle sezioni composte da 170. Per evitare di doverli ribattere, potete utilizzare il comando Nuovo indirizzo per saltare il blocco di 170. Siate, comunque, prudenti; non dovete saltare qualcosa che *dovreste* in realtà battere.

Potete utilizzare i comandi di Registrazione e Lettura per ottenere delle copie del programma completo. Usate il comando di Lettura per ricaricare il programma da nastro o da disco, quindi inserite un nuovo nastro o disco ed usate il comando di Registrazione per creare una nuova copia.

Un inconveniente che si presenta con la registrazione su nastro: quando caricate da nastro il messaggio «FOUND nome del programma» può apparire due volte. L'unità nastri, tuttavia, funziona correttamente.

I programmatori troveranno MLX molto interessante in termini di protezione dell'utente da errori. Ci sono anche particolari interessanti riguardo la predisposizione dello schermo. Particolarmente rilevante è l'uso dei sottoprogrammi della ROM Kernal per leggere e registrare blocchi di memoria. Limitatevi ad inserire con una POKE l'indirizzo di partenza (suddiviso in byte meno e più significativo) nelle locazioni 251 e 252 e l'indirizzo finale in 254 e 255. Il codice che contraddistingue un qualsiasi errore può essere letto nella locazione 253 (un errore verrà segnalato da un numero di codice minore di dieci).

Mi auguro che consideriate MLX un programma che permette un effettivo risparmio di tempo. Dal momento che è stato collaudato copiando dei veri programmi, potete fidarvene come di un programma in grado di generare linguaggio macchina esente da errori.

Appendice I

```
100 PRINT"{CLR}{CYN}";CHR$(142);CHR$(8);:P
    OKE53281,1:POKE53280,1
101 POKE788,52:REM DISABILITA RUN/STOP
110 PRINT"{RVS}{ 40 SPAZI}";
120 PRINT"{RVS}{ 15 SPAZI}{DES}{OFF}[<*>]
    £{RVS}{DES} {DES}{ 2 SPAZI}[<*>]{OFF}
    [<*>]£{RVS}£{RVS}{ 13 SPAZI}";
130 PRINT"{RVS}{ 15 SPAZI}{DES}[<N>][<H>]
    {DES} { 2 DES} {OFF}£{RVS}£[<*>]{OFF}
    [<*>]{RVS}{ 13 SPAZI}";
140 PRINT"{RVS}{ 40 SPAZI}"
200 PRINT"{ 2 GIU'}{PUR} UN PROGRAMMA
    { 3 SPAZI}PER{ 3 SPAZI}L'INTRODUZIONE
    DI{ 2 SPAZI}ROUTINE IN LINGUAGGIO";
205 PRINT" MACCHINA A PROVA{ 17 SPAZI}DI
    { 2 SPAZI}ERRORE{ 3 GIU'}"
210 PRINT"[<5>]{ 2 SU} INDIRIZZO DI PARTEN
    ZA{ 2 SPAZI}";:INPUTS:F=1-F:C$=CHR$(31
    +119*F)
220 IFS<256OR(S>40960ANDS<49152)ORS>53247T
    HENGOSUB3000:GOTO210
225 PRINT:PRINT:PRINT
230 PRINT"[<5>]{ 2 SU} INDIRIZZO CONCLUSIV
    O{ 3 SPAZI}";:INPUTE:F=1-F:C$=CHR$(31+
    119*F)
240 IFE<256OR(E>40960ANDE<49152)ORE>53247T
    HENGOSUB3000:GOTO230
250 IFE<STHENPRINTC$;"{RVS}INDIRIZZO CONCL
    USIVO<INDIRIZZO INIZIALE"
255 IFE<STHENGOSUB1000:GOTO230
260 PRINT:PRINT:PRINT
300 PRINT"{CLR}";CHR$(14):AD=S:POKEV+21,0
310 PRINTRIGHT$("0000"+MID$(STR$(AD),2),5)
    ;":";:FORJ=1TO6
320 GOSUB570:IFN=-1THENJ=J+N:GOTO320
390 IFN=-211THEN710
400 IFN=-204THEN790
410 IFN=-206THENPRINT:INPUT"{GIU'} INSERIR
```

```
    E IL NUOVO INDIRIZZO";ZZ
414 IFN=-206THENIFZZ<SORZZ>ETHENPRINT"
    {RVS}ESCE DAL CAMPO DI VALORI INDICATO
    "
415 IFN=-206THENIFZZ<SORZZ>ETHENGOSUB1000:
    GOTO410
417 IFN=-206THENAD=ZZ:PRINT:GOTO310
420 IFN<>-196THEN480
430 PRINT:INPUT"LISTATO:DA";F:PRINT"
    { 9 SPAZI}A";:INPUTT
440 IFF<SORF>EORT<SORT>ETHENPRINT"MINIMO";
    S;" MASSIMO";E;"![<5>]":GOTO430
450 FORI=FTOTSTEP6:PRINT:PRINTRIGHT$("0000
    "+MID$(STR$(I),2),5);":";
451 FORK=0TO5:N=PEEK(I+K):PRINTRIGHT$("00"
    +MID$(STR$(N),2),3);",";
460 GETA$:IFA$>""THENPRINT:PRINT:GOTO310
470 NEXTK:PRINTCHR$(20);:NEXTI:PRINT:PRINT
    :GOTO310
480 IFN<0THENPRINT:GOTO310
490 A(J)=N:NEXTJ
500 CKSUM=AD-INT(AD/256)*256:FORI=1TO6:CKS
    UM=(CKSUM+A(I))AND255:NEXT
510 PRINTCHR$(18);:GOSUB570:PRINTCHR$(20)
515 IFN=CKSUMTHEN530
520 PRINT:PRINT"{RED}LA LINEA E' STATA INS
    ERITA IN MANIERA"
525 PRINT"ERRATA. RIPETERE[<5>]":PRINT:GOS
    UB1000:GOTO310
530 GOSUB2000
540 FORI=1TO6:POKEAD+I-1,A(I):NEXT:POKE542
    72,0:POKE54273,0
550 AD=AD+6:IFAD<ETHEN310
560 GOTO710
570 N=0:Z=0
580 PRINT"[<+>]";
581 GETA$:IFA$=""THEN581
585 PRINTCHR$(20);:A=ASC(A$):IFA=13ORA=44O
    RA=32THEN670
```

```
590 IFA>128THENN=-A:RETURN
600 IFA<>20THEN630
610 GOSUB690:IFI=1ANDT=44THENN=-1:PRINT"
    {SIN} {SIN}";:GOTO690
620 GOTO570
630 IFA<48ORA>57THEN580
640 PRINTA$;:N=N*10+A-48
650 IFN>255THENA=20:GOSUB1000:GOTO600
660 Z=Z+1:IFZ<3THEN580
670 IFZ=0THENGOSUB1000:GOTO570
680 PRINT",";:RETURN
690 S%=PEEK(209)+256*PEEK(210)+PEEK(211)
691 FORI=1TO3:T=PEEK(S%-I)
695 IFT<>44ANDT<>58THENPOKES%-I,32:NEXT
700 PRINTLEFT$("{ 3 SIN}",I-1);:RETURN
710 PRINT"{CLR}{RVS}*** SAVE ***{ 3 GIU'}
    "
720 INPUT"{GIU'}NOME DEL FILE";F$
730 PRINT:PRINT"{ 2 GIU'}{RVS}N{OFF}ASTRO
    O {RVS}D{OFF}ISCO: (N/D)"
740 GETA$:IFA$<>"N"ANDA$<>"D"THEN740
750 DV=1-7*(A$="D"):IFDV=8THENF$="0:"+F$
760 T$=F$:ZK=PEEK(53)+256*PEEK(54)-LEN(T$)
    :POKE782,ZK/256
762 POKE781,ZK-PEEK(782)*256:POKE780,LEN(T
    $):SYS65469
763 POKE780,1:POKE781,DV:POKE782,1:SYS6546
    6
765 POKE254,S/256:POKE253,S-PEEK(254)*256:
    POKE780,253
766 POKE782,E/256:POKE781,E-PEEK(782)*256:
    SYS65496
770 IF(PEEK(783)AND1)OR(ST AND191)THEN780
775 PRINT"{GIU'}FATTO.":END
780 PRINT"{GIU'}ERRORE NEL SAVE-RIPROVA!":
    IFDV=1THEN720
781 OPEN15,8,15:INPUT#15,E1$,E2$:PRINTE1$;
    E2$:CLOSE15:GOTO720
790 PRINT"{CLR}{RVS}*** LOAD ***{ 2 GIU'}
```

```
    "
800 INPUT"{ 2 GIU'}NOME DEL FILE";F$
810 PRINT:PRINT"{ 2 GIU'}{RVS}N{OFF}ASTRO
    O {RVS}D{OFF}ISCO: (N/D)"
820 GETA$:IFA$<>"N"ANDA$<>"D"THEN820
830 DV=1-7*(A$="D"):IFDV=8THENF$="0:"+F$
840 T$=F$:ZK=PEEK(53)+256*PEEK(54)-LEN(T$)
    :POKE782,ZK/256
841 POKE781,ZK-PEEK(782)*256:POKE780,LEN(T
    $):SYS65469
845 POKE780,1:POKE781,DV:POKE782,1:SYS6546
    6
850 POKE780,0:SYS65493
860 IF(PEEK(783)AND1)OR(ST AND191)THEN870
865 PRINT"{GIU'}FATTO.":GOTO310
870 PRINT"{GIU'}ERRORE NEL LOAD-RIPETI!
    {GIU'}":IFDV=1THEN800
880 OPEN15,8,15:INPUT#15,E1$,E2S:PRINTE1$;
    E2$:CLOSE15:GOTO800
1000 REM CICALINO
1001 POKE54296,15:POKE54277,45:POKE54278,1
     65
1002 POKE54276,33:POKE54273,6:POKE54272,5
1003 FORT=1TO200:NEXT:POKE54276,32:POKE542
     73,0:POKE54272,0:RETURN
2000 REM CAMPANELLO
2001 POKE54296,15:POKE54277,0:POKE54278,24
     7
2002 POKE54276,17:POKE54273,40:POKE54272,0

2003 FORT=1TO100:NEXT:POKE54276,16:RETURN
3000 PRINTC$;"{RVS} NON IN PAGINA ZERO O S
     U{DES}ROM ":GOTO1000
```

TERZA PARTE

IL SISTEMA OPERATIVO GEOS

IL GEOS

Il sistema operativo GEOS è un programma orientato verso l'utente, in quanto la comunicazione con il Commodore 64 è stata notevolmente semplificata.
Per arrivare a tale risultato, gran parte del sistema operativo residente nel computer (Kernal), è stato riscritto; inoltre si è dovuto aumentare notevolmente la velocità di comunicazione con il drive 1541 (4-5 volte), in quanto altrimenti non si sarebbe potuto ottenere un risultato commercialmente apprezzabile.
In tal modo è stato realizzato un programma estremamente professionale che garantisce un ancor brillante futuro al Commodore 64.
GEOS è un vero e proprio ambiente di lavoro, in quanto le applicazioni inserite possono trasmettersi facilmente testi e immagini, risolvendo semplicemente un vecchio problema che tanti grattacapi ha generato: quello della compatibilità.
Il dischetto che avete ricevuto è detto disco di sistema GEOS: tenetelo con cura in quanto è l'unico con cui potrete caricare tutto il sistema.
Questa appendice è stata scritta in modo estremamente semplice, onde facilitare l' approccio con GEOS. Noi vi consigliamo di leggerla con il vostro Commodore di fianco: potrete così esercitarvi mano a mano che vi illustreremo gli strumenti e i comandi. Vi sarà così più immediato il significato delle varie procedure.
A una generica introduzione agli strumenti del GEOS, seguirà una dettagliata relazione su GEOPAINT, GEOWRITE e sugli accessori che avrete a disposizione.

CAPITOLO 1

COMANDI FONDAMENTALI

Cominciamo subito a utilizzare il sistema operativo GEOS; prendete il dischetto “GEOS”, inseritelo nel drive e impostate i comandi seguenti:

```
load “geos”,8
run
```

Dopo qualche secondo apparirà l’immagine in fig. 1.1. Questa rappresenta una specie di menu principale da cui si può:

— accedere alle varie applicazioni che lavorino in ambiente GEOS; sul vostro dischetto ne trovate due GEOPAINT e GEOWRITE

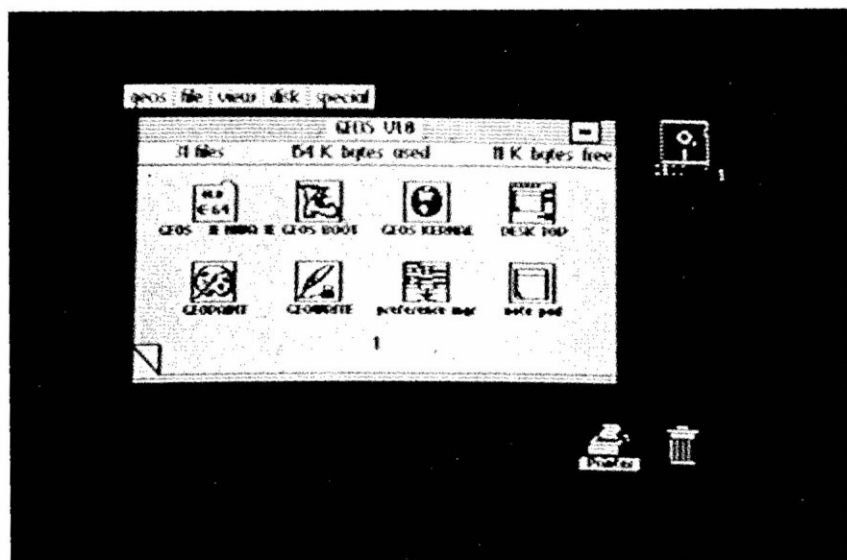



Figura 1.1 - Schermata relativa al Desktop.

— accedere a programmi Basic scritti da voi
— accedere a strumenti ausiliari messi a disposizione dal GEOS, e cioè una calcolatrice ,un blocco di appunti e così via
— infine potrete gestire i file contenuti nel dischetto inserito cancellandoli, copiandoli e così via.

Questa schermata iniziale viene chiamata DESKTOP e verrà richiamata ogni volta che uscirete da una applicazione.
A questo punto inserite il joystick nella porta di controllo 1, e provate a muovere la manopola: la freccia visibile sullo schermo seguirà i vostri movimenti. Essa è l'unico modo per comunicare al GEOS quale operazione effettuare; il suo uso è però molto semplice e intuitivo.
Anzitutto provate a muoverla verso la parte superiore sinistra dello schermo e puntatela sulla parola "geos"; quindi schiacciate il tasto di fire e vedrete apparire il menu rappresentato in fig. 1.2 (pull-down menu). Una volta attivato tale menu si può selezionare l'opzione voluta semplicemente muovendo la freccia sulla parola corrispondente e schiacciando il pulsante di fire; il menu viene così pure disattivato.

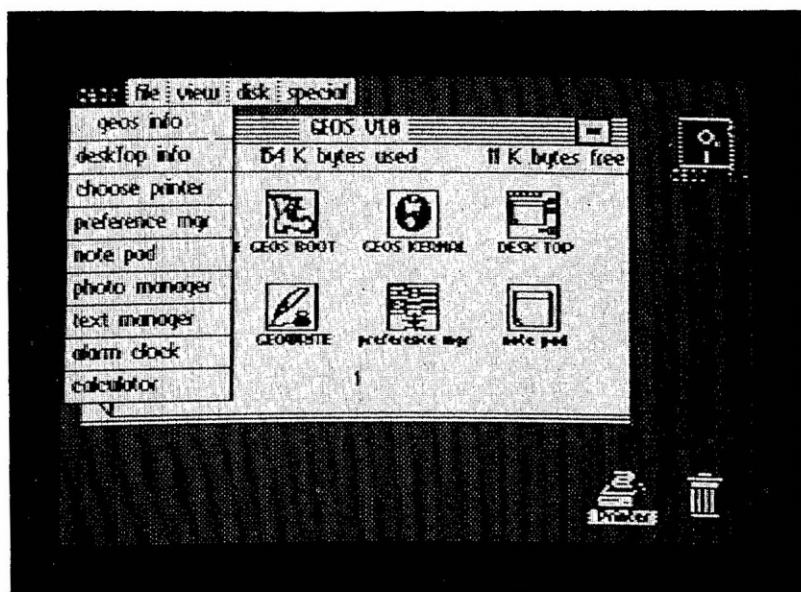



Figura 1.2 - Pull-down menu in funzione.

Ad esempio provate a selezionare l'opzione "geos info": vi compariranno una serie di informazioni sul GEOS. Per ritornare al desktop schiacciate il tasto di fire.
Abbiamo così imparato.ad attivare i menu superiori.

Passiamo ora all'immagine centrale sullo schermo: questa rappresenta una finestra, cioè una zona grafica dedicata dal GEOS al programma attualmente attivo. Questo significa che ogni volta che è necessario visualizzare dei messaggi, dei testi o delle immagini, il sistema operativo deve prima riservare una zona dello schermo: lì verrà svolto tutto il lavoro.
In questo caso la finestra rappresenta il dischetto in uso, informandovi sul suo nome, sullo spoazio disponibile e così via.
Nella parte superiore si trova l'intestazione, cioè il nome del disco attualmente inserito; in questo caso GEOS più la versione. Sulla stessa riga ma all'estrema destra si trova un quadrato utile per chiudere il disco attualmente inserito nel drive.
Sulla linea inferiore si trovano informazioni relative al disco presente, e cioè il numero di file, la memoria occupata e quella libera.
Al disotto si trova un cosidetto taccuino del direttorio su cui sono segnati i vari file presenti sul dischetto.
Tale denominazione è giustificata dal fatto che si dispongono di al massimo di otto pagine il cui numero è segnato in basso al centro. Queste possono essere fatte scorrere puntando la freccia sull'angolo in basso a sinistra e schiacciando il bottone di fire; selezionando la parte superiore o inferiore dell'angolo si otterrà la visualizzazione della pagina seguente o precedente.
I file sono rappresentati mediante immagini dette icone. Queste rappresentano indicativamente il tipo di lavoro svolto dall'applicazione; su una pagina ne sono rappresentate otto. Le icone possono essere "attivate": basta puntare sopra la freccia e schiacciare il bottone di fire; noterete il cambiamento di colore dell'immagine: ciò è caratteristico di ogni attivazione. A questo punto opzioni selezionate dai menu superiori riguarderanno il file prescelto.

Schiacciando il bottone di fire due volte di seguito, lasciando in mezzo una pausa di un secondo, apparirà una immagine ulteriore rappresentante i contorni dell'icona; tale elemento potrà essere "trascinato" in giro per lo schermo e rilasciato premendo il tasto di fire. Lo scopo di tale operazione è generalmente quello di copiare un file, di cancellarlo portandolo nel cestino o di stamparlo portandolo sulla stampante. Vedremo in seguito in dettaglio come effettuare tali operazioni.
Provate quindi a cancellare il file "GEOWRITE". Il GEOS vi comunicherà che tale operazione non è possibile in quanto esso è protetto.

Per ritornare al Desktop puntate la freccia sulla casella contenente la scritta O.K. e schiacciate il tasto di fire. Quest'ultimo modo di selezionare opzioni è molto frequente. In particolare vi appariranno dei riquadri in cui vengono presentate alcuni possibili scelte (riquadri di dialogo) : voi puntate su quella che vi interessa e azionate il tasto di fire.

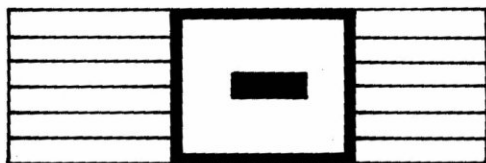



Figura 1.3 - Particolare della finestra contenente il taccuino del direttorio; posizionando la freccia sul quadratino e premendo il tasto di fire, potrete chiudere il disco.

Ad esempio provate a schiacciare due volte di seguito, cioè senza pausa, sull'icona GEOWRITE; in tal modo potrete accedere al wordprocessor, un'applicazione contenuta sul disco.Questa operazione viene chiamata doppio click. Dopo alcuni secondi vi si presenteranno tre scelte:
— creare un nuovo documento
— aprirne uno già esistente
— ritornare al Desktop (quit)
Per il momento scegliete l'opzione QUIT: di Geowrite ci occuperemo in seguito.
Analizziamo ora l'ultima parte del Desktop relativa alla apertura e chiusura di dei dischetti. Provate a puntare la freccia sul quadratino in alto a destra dell'intestazione della finestra (GEOS più la versione in uso) e schiacciate il tasto di fire (fig 1.3). La finestra si sbiancherà e sul dischetto rappresentato in alto a destra apparirà un punto interrogativo.
Questo significa che il GEOS attende l'inserimento di un nuovo dischetto avendo, per così dire, chiuso quello attualmente inserito. In particolare la zona di memoria contenente tutte le informazioni relative, è stata cancellata e i canali di comunicazione con il drive interrotti.
L'operazione inversa è denominata "apertura" del disco, e procede a caricare tutti i dati ad esso relativi.
Una volta inserito nel drive il dischetto nuovo, è sufficente puntare sull'imagine del floppy in alto a destra e schiacciare il tasto di fire.
La sequenza di apertura e chiusura è necessaria ogni qualvolta si vuole cambiare disco, in modo che il Desktop possa essere aggiornato.
Proviamo ora a inserire al posto del disco GEOS, uno qualunque contenente i vostri programmi scritti in basic.

Dopo averlo aperto, appare una finestra indicante che questo è scritto in un formato non GEOS; l'alternativa è fra renderlo compatibile oppure no. Selezionando la scelta YES farete sì che il file del direttorio venga allungato di un blocco, senza per questo perdere alcun dato.
Ciò non ha molta importanza e potrete comunque scegliere l'opzione "NO" che non modificherà il contenuto del disco.
A questo punto appaiono sulla finestra centrale le icone dei vostri programmi; questi possono essere eseguiti semplicemente operando un doppio click sull'icona scelta.
Potrete così vedere funzionare il vostro programma; per ritornare al desktop inserite il disco del GEOS e schiacciate contemparaneamente i tasti RUN-/STOP e RESTORE.

In seguito useremo la dizione "aprire", anche per i programmi e le applicazioni, intendendo così l'operazione di accesso a questi. Con "chiudere" intenderemo il ritorno al Desktop.

Abbiamo così visto in questo capitolo come usare il joystick per selezionare le operazioni volute. Rimane soltanto da mettere in evidenza la semplicità d'uso di un tale sistema, che rende accessibile a tutti l'uso del computer.

Riassunto delle azioni fondamentali del joystick

Attivazione di un menu superiore (pull down-menu)

— Puntate la freccia su uno dei riquadri superiori e azionate il tasto di fire; apparirà così il menu contenente le opzioni possibili
— Desiderando un menu diverso, è sufficiente portare la freccia al di fuori del medesimo; verrà così disattivato

Selezione di un'opzione da un menu superiore

— Dopo avere attivato il menu superiore corretto, puntate la freccia sull'opzione che vi interessa e premete il tasto di fire

Attivazione di un file

— Puntate la freccia sull'icona del file prescelto e schiacciate il tasto di fire; essa cambierà così colore
— Desiderando un file diverso, portate la freccia al di fuori dell'icona attivata e premete il tasto di fire

Trascinamento di un file

— Puntate la freccia sull'icona del file prescelto e schiacciate due volte di seguito, lasciando una certa pausa, il tasto di fire; in tal modo prima si attiva il file e in seguito apparirà un'immagine supplementare che potrà essere portata ovunque sullo schermo
— Una icona così trascinata può essere depositata nel luogo voluto semplicemente premendo il tasto di fire, quando la sua immagine ha assunto la posizione voluta

Finestre supplementari

Molto spesso il GEOS vi offrirà ,in una finestra supplementare, una serie di possibili alternative: per sceglierne una, puntate la freccia sul riquardo contenente quella che vi interessa e premete il tasto di fire.
Ecco il significato di alcune scritte spesso ricorrenti.

QUIT: se siete nei pasticci (vi siete "persi") questa vi permetterà di tornare al Desktop

O.K.: il GEOS vi sta visualizzando un messaggio (in genere di errore) e per proseguire dovete scegliere tale opzione; a volte può pure indicare una risposta positiva a una domanda

CANCEL: annulla l'operazione in corso e ritorna al Desktop; anche questa può diventare un'ancora di salvezza.

Alcune volte il GEOS vi chiederà di inserire il nome di un file: questo è l'unico caso in cui dovrete usare la tastiera e non dimenticate di premere il tasto di RETURN quando avete finito.

Doppio click

Azione destinata ad aprire un file senza passare per i menu superiori.

— Puntate la freccia sull'icona del file che vi interessa e premete il tasto di fire due volte rapidamente, cioè senza lasciare una pausa;
in caso contrario non fareste che iniziare una operazione di trascinamento: per correggere è sufficiente che premiate il tasto di fire un'altra volta in modo da lasciare tutto come in precedenza.

1.2 I comandi del Desktop in dettaglio

Apertura e chiusura del disco

Un disco può essere aperto in due modi:

— selezionando il dischetto in alto a sinistra
— attivando il menu "disk" e selezionando l'opzione open

In tal modo la finestra centrale viene riempita con le icone corrispondenti ai file contenuti nel direttorio.
Se il disco inserito non fosse stato creato con il GEOS, vi verrà chiesto se renderlo GEOS compatibile; tale operazione costa un blocco (256 bytes) sul dischetto e nessun dato sarà cancellato.
Per chiudere un dischetto si può operare in due modi:

— puntando sul quadratino in alto a destra e azionando il tasto di fire
— selezionando con il menu "disk" l'opzione "close".

Tali operazioni permettono di cambiare il dischetto contenuto nel drive.

Comandi per la gestione del dischetto

Attivando il menu "disk" oltre alle oprazioni di close e open si hanno a disposizione molte altre possibilità:

RENAME : permette di cambiare il nome del disco attualmente aperto;

vi verrà chiesto il nuovo nome del disco: inseritelo e premete il tasto RETURN.

FORMAT : serve a formattare un dischetto;
- prima di attivare il menu, chiudete il disco attualmente in uso e inserite quello da formattare
- attivate il menu "disk", selezionate l'opzione format
- il GEOS vi chiederà di inserire il nome del disco: fatelo mediante la tastiera, senza dimenticare di premere RETURN alla fine

VALIDATE: serve a recuperare quei blocchi che vengono inutilmente mantenuti occupati

ADD DRIVE: permette di aggiungere un drive supplementare B

COPY: permette di copiare il disco attualmente aperto su un disco destinazione, precedentemente formattato; in particolare si può creare un backup di tutto il disco GEOS

Quest'ultima operazione è particolarmente importante in quanto è sempre presente la possibilità di perdere dei programmi sul dischetto GEOS; pertanto vi consigliamo caldamente di creare un disco di backup. Questo però non potrà essere usato per caricare il GEOS e vi servirà come serbatoio di programmi.
Prendete un disco vuoto e formattatelo scegliendo l'opzione "format" dal menu "disk"; scegliete un nome qualunque ma che non sia "GEOS". Terminata la formattazione reinserite il dischetto originale, apritelo (come spiegato all'inizio di questo paragrafo) e selezionate dal menu "disk" l'opzione "copy" ; il programma vi chiederà alternativamente di inserire il dischetto di backup e quello originale finchè la copia non sarà terminata.
A questo punto prendete il disco di backup e mettetelo in un luogo sicuro: se un programma del disco GEOS non dovesse più funzionare lo sostituirete con quello equivalente seguendo le indicazioni del prossimo paragrafo.

Copia di un file

Se volete duplicare un file sullo stesso disco, seguite la procedura seguente:

— attivate l'icona del file interessato
— attivate il menu "file" e selezionate l'opzione "duplicate"
— il GEOS vi chiederà il nuovo nome di questo file, in quanto non ne possono coesistere due sullo stesso dischetto con lo stesso nome; inseritelo mediante la tastiera, non dimenticando di schiacciare il tasto RETURN alla fine.

Per duplicare un file da un disco all'altro dovete:

— trascinare l'icona del file interessato al di fuori del direttorio portandolo nella parte inferiore dello schermo, e depositarla lì

— estrarre dal drive il disco sorgente
— inserire quello destinazione e aprirlo
— trascinare l'icona del file che si trova in basso, dentro il taccuino del direttorio e depositarla nel punto voluto
— seguire le indicazioni del GEOS che vi chiederà di inserire alternativamente il disco sorgente e quello destinazione

Ovviamente il disco destinazione dovrà essere GEOS compatibile. Siccome si possono depositare fino a quattro icone sul fondo, questo è il numero massimo di file che potrete copiare in una volta.

Come mettere in ordine il taccuino del direttorio

Abbiamo visto che il direttorio è rappresentato da un taccuino di al massimo otto pagine su cui sono segnate le icone dei file presenti; queste possono essere spostate a piacere in modo da avere una visualizzazione più vicina alle esigenze dell'utente. Per muovere un'icona sul taccuino dovete:

— trascinare quella del file interessato sul fondo, sotto il taccuino e rilasciarla lì
— selezionare la pagina del taccuino su cui dovrà essere inserita l'icona
— trascinarla all'interno del taccuino, rilasciandola nel punto desiderato.

In tal modo potrete separare i vari file del direttorio, mantenendo il taccuino in ordine.

Comandi per gestire i file

Attivando il menu "file" appaiono alcune opzioni necessarie per gestire i file presenti sul dischetto attualmente aperto.
Prima di selezionarle è però necessario attivare l'icona del file su cui si vuole agire.

OPEN: apre il file attivato; è equivalente al doppio click

DUPLICATE: copia il file attivato sullo stesso disco; il Geos vi chiederà di introdurre un nome nuovo.

RENAME: permette di cambiare il nome al file attivato PRINT : stampa il file attivato; è equivalente a trascinare e rilasciare l'icona sulla stampante; per scegliere la corretta stampante leggere il capitolo dedicato agli accessori del GEOS

GET INFO: visualizza informazioni varie sul file attivato;

Per quanto riguarda quest'ultima opzione, bisogna notare la presenza, sotto la scritta "author", di un quadratino che indica se un file è protetto oppure no (se si ha lo stesso colore dei caratteri) ; può essere cambiato semplicemente puntando la freccia sopra e premendo il tasto di fire. Questo vi permetterà di proteggere i vostri documenti da cancellazioni accidentali, oppure di sproteggere i programmi GEOS: quest'ultima è un'attivtà sconsigliata in quanto potreste cancellare applicazioni importanti come GEOWRITE.
In fondo alla finestra esiste un riquadro su cui potrete scrivere i vostri commenti usando la tastiera.
Per tornare al Desktop dovete togliere la finestra: puntate la freccia sul quadratino in alto a destra della stessa; (a lato del nome del file su cui si leggono le informazioni) e premete il tasto fire.

Visualizzazione del direttorio

Attivando il menu "view" è possibile visualizzare il direttorio con un elenco di nomi in vece che di icone. Bisogna perciò selezionare con quale criterio verranno ordinati i file.

NAME: i file vengono presentati in ordine alfabetico

DATE: vengono presentati in ordine temporale; il file aggiornato più recentemente è scritto per primo

SIZE: vengono presentati in ordine di grandezza, il più grande viene prima

KIND: vengono presentati raggruppati per tipo

È importante notare che un file può essere attivato se e solo se è presente sullo schermo il taccuino del direttorio con le icone.
Per visualizzarlo è sufficiente attivare il menu "view" e selezionare l'opzione "icon".

Ritorno al Basic

Attivando il menu "special" e selezionando l'opzione "BASIC" tornerete al basic, dove potrete scrivere i vostri programmi normalmente. Per ritornare al GEOS dovete:

— inserire nel drive il dischetto GEOS
— premere contemporaneamente i tasti RUN/STOP e RESTORE

Altre opzioni del menu "special" sono:

— RESET ricarica il GEOS

CAPITOLO 2

GEOPAINT

2.1 INTRODUZIONE

GEOPAINT è un potente e flessibile editor grafico che semplifica la stesura di documenti in cui l'uso delle figure è predominante rispetto a quello del testo. Quest'editor, come ogni parte del sistema operativo GEOS, è orientato all'utente finale che non abbia familiarità con alcun sistema di calcolo. Per questo è stato riprodotto un ambiente di lavoro simile a quello in cui ogni disegnatore è abituato a svolgere la propria attività. Ci si troverà così a lavorare con carta, matita, gomma e qualsiasi altro strumento che ogni grafico ha a disposizione.

2.2 IL PRIMO DOCUMENTO CON GEOPAINT

L'ambiente che si incontra dopo aver lanciato il sistema operativo GEOS mostra i file presenti sul dischetto mediante delle icone a cui è associato il nome del file che esse rappresentano. Fra le tante mostrate, vi sarà anche quella su cui appare il disegno di una tavolozza di colori. Questa è l'icona di GEOPAINT. Aprendo con un doppio click questa applicazione, si entrerà nell'ambiente di lavoro più sopra illustrato. Dopo qualche secondo vi si mostrerà la schermata visibile in fig. 2.1. Poichè finora non avete ancora lavorato con GEOPAINT, e quindi non disponete di un documento preesistente, dovrete scegliere l'opzione CREATE che vi mette a disposizione un nuovo foglio da disegno su cui lavorare. Questo sarà in futuro rintracciabile mediante il nome che ora vi viene richiesto da tastiera e che dovete terminare con il tasto RETURN. Se durante la battitura avete commesso qualche errore, potrete correggerlo col tasto INST/DEL purchè non abbiate ancora digitato il tasto RETURN. L'opzione CANCEL che avete sul video non serve per correggere il nome del file, ma provvede ad interrompere l'operazione CREATE che avete intrapreso.
Ora vi trovate al tavolo di disegno con un foglio di carta bianca davanti. Sulla sinistra trovate gli strumenti da disegno che avete a disposizione. Di questi potrete usarne uno solo per volta e quello che adopererete sarà evidenziato

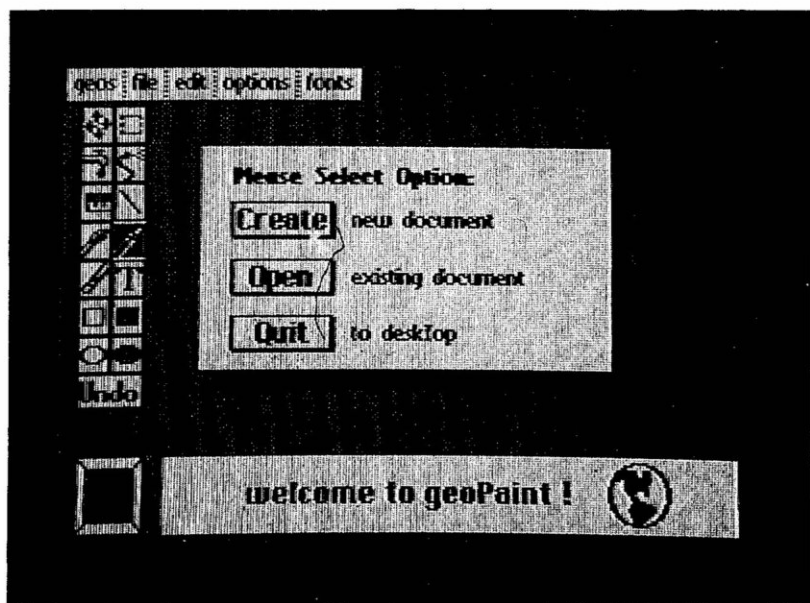



Figura 2,1 - L'editor grafico GEOPAINT.

con un colore di fondo diverso. Poichè siete appena entrati in questo ambiente si assume che stiate usando la MATITA. Questa è d'uso estremamente semplice: se vi trovate già sul foglio da disegno, per iniziare il tracciamento di una linea sarà sufficiente premere il tasto di fire per appoggiare la punta della MATITA sul foglio e quindi muoversi mediante il joystick nella direzione voluta. Azionando nuovamente il tasto di fire, alzeremo la matita dal foglio di disegno e ciò ci permetterà di spostarci liberamente. Come vi sarete accorti il colore della matita cambia a seconda che questa sia appoggiata o sollevata dal foglio.

Per posare lo strumento che abbiamo in mano e prendere quello desiderato, è sufficiente spostare la freccia su quest'ultimo e premere il tasto di fire. Per ora non faremo niente di simile ma ci accontenteremo di imparare ad uscire da questo ambiente per ritornare al DESKTOP. Ciò si otterrà attivando il pull-down menu che contiene la scritta FILE e quindi selezionando l'opzione QUIT. Ora siamo ritornati al DESKTOP e sfogliando il taccuino del diretto-rio troveremo una nuova icona corrispondente al documento su cui abbiamo tracciato una linea.

2.3 GLI STRUMENTI DI GEOPAINT IN DETTAGLIO

MOVIMENTO DELL'AREA GRAFICA

Il foglio su cui lavoriamo è molto più grande di quanto possiamo vedere sul nostro schermo. La parte che ci viene mostrata è detta AREA GRAFICA. Per conoscerne la posizione, dobbiamo guardare nella finestra di stato (STATUS WINDOW) posta nell'angolo in basso a destra del video. Infatti vi troviamo rappresentati sia il foglio da disegno (rettangolo grande) che l'AREA GRAFICA stessa (rettangolo piccolo). Per visualizzare una diversa porzione del nostro documento, dovremo spostare la posizione di quest'ultima mediante lo strumento di lavoro che in fig. 2.2 è indicato con 6. Dopo averlo attivato appariranno in sovraimpressione 4 frecce fra loro ortogonali che indicano i quattro possibili spostamenti selezionabili con il joystick. La finestra di stato ci mostrerà ad ogni istante la posizione dell'AREA GRAFICA all'interno del foglio. Premendo il tasto di fire questo strumento viene deposto e viene riattivato l'ultimo strumento usato.

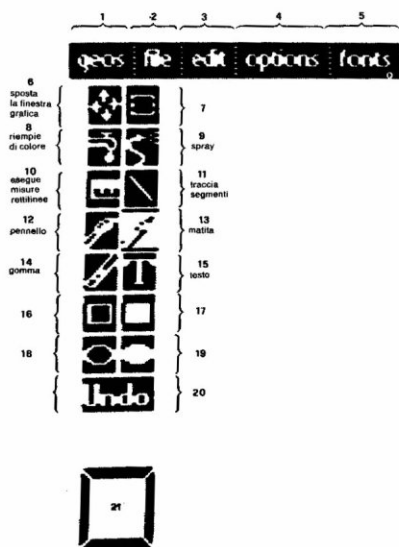



Figura 2.2 - Gli strumenti di lavoro di GEOPAINT.

BOX

Lo strumento, che in fig. 2.2 è indicato con 7, ci permette di definire un rettangolo (BOX) all'interno dell'AREA GRAFICA. Questo ha i lati paralleli al bordo del foglio e pertanto è individuabile con gli estremi di una sua diagonale. La parte di disegno che vi è contenuta può essere mossa, copiata, cancellata e manipolata secondo le varie opzioni che vengono presentate nella finestra di stato all'atto dell'attivazione di BOX.
Vediamone l'uso: dopo averlo attivato, bisogna fissare la posizione e le dimensioni del rettangolo definendo una sua diagonale. Ciò si realizza in due passi:

a) ci si posiziona sul primo estremo di questa e si preme il tasto fire
b) si ripete l'operazione precedente anche per il secondo estremo.

Fatto ciò possiamo selezionare nella finestra di stato l'opzione desiderata. Se questa dovesse essere COPY o MOVE allora dovremo riposizionarci sull'ultimo estremo della diagonale definita e premere il tasto di fire per agganciare alla freccia il rettangolo. Ora, senza fermarsi, si sposterà la freccia nella nuova posizione che si desidera che BOX assuma. Appena la freccia si sarà arrestato, il contenuto di BOX verrà spostato o copiato nella nuova posizione. Premendo nuovamente il tasto di fire si sgancerà la freccia da BOX. Diversamente ogni movimento della freccia verrà trasmesso a BOX.

RUBINETTO

Questo strumento, indicato con 8 in fig. 2.2, provvede a riempire un'area chiusa da linee con il colore correntemente selezionato dalla tavolozza dei colori (vedi). Se viene attivato su un'area aperta l'intera AREA GRAFICA verrà colorata.
Per usare RUBINETTO sarà sufficiente attivarlo, portarsi all'interno dell'area chiusa e premere il tasto di fire.

SPRAY

Questo strumento, indicato in fig. 2.2 con 9, permette di disegnare a spruzzo. L'uso di SPRAY è del tutto simile a quello della MATITA (vedi) ma differisce per il tipo di tratto che realizza.

RIGHELLO

Talvolta sorge la necessità di misurare la distanza fra due punti. RIGHELLO (punto 10 in fig. 2.2) soddisfa a questa necessità.
Per usare questo strumento è necessario che lo selezionate e che vi posizionate su uno dei due punti di cui volete conoscere la distanza. Premete il tasto di fire, posizionatevi sul secondo punto e premetelo nuovamente. La misura desiderata, insieme ad altre informazioni, è mostrata nella finestra di stato. È possibile ottenere questa misura in pixel o in pollici semplicemente attivando nella finestra di stato l'unità di misura desiderata.

TRACCIATORE DI SEGMENTI

Come vi sarete accorti, tracciare con la MATITA delle righe diritte con un'inclinazione qualsiasi, non è un'operzione agevole.
Il TRACCIATORE DI SEGMENTI, che in fig. 2.2 è indicato con 11, permette di risolvere rapidamente questo problema. L'uso di questo strumento è simile al precedente (vedi).

PENNELLO

Come abbiamo visto più sopra, la MATITA ci consente di tracciare con la massima libertà una generica linea. Purtroppo il tratto di questa è fisso.
Il PENNELLO, pur mantenendo le stesse potenzialità e modalità d'uso di MATITA (vedi), ci consente di selezionare in un vasto insieme di tratti predefiniti quello più adatto al disegno. Per questo vi si rimanda al paragrafo in cui vengono trattate le opzioni.

MATITA

Questo strumento, indicato con 13 in fig. 2.2, permette tracciare con la massima libertà delle linee sottili. Se desiderate utilizzarlo selezionatelo e spostatevi nel punto in cui dovete iniziare a tracciare.
Premete il tasto di fire e quindi spostatevi utilizzando il joystick. La punta della matita lascerà un sottile tratto dove passerà. Se desiderate sollevare la matita dal foglio basterà agire nuovamente sul tasto di fire.

GOMMA

Questo strumento (fig. 1 punto 14) permette di cancellare qualsiasi segno indesiderato.
L'uso della GOMMA è simile a quello di MATITA (vedi).

TRASFERIBILI

Talvolta sorge la necessità di corredare un disegno con un testo. I TRASFERIBILI, che in fig. 2.2 sono indicati con 15, soddisfano a questa esigenza. I caratteri disponibili appartengono a diverse serie (FONTS).
Queste possono essere modificate sia selezionando una particolare dimensione del carattere sia utilizzando uno dei vari stili scelto fra quelli disponibili (per esempio bordato, sottolineato, ...). La selezione di una serie di caratteri e delle loro dimensioni sarà trattata nel paragrafo FONTS.
Qui viene solo spiegato l'uso dei TRASFERIBILI e le modalità di scelta di uno stile. Per aggiungere del testo ad un disegno è necessario anzitutto attivare i TRASFERIBILI e successivamente definire il rettangolo, detto FINESTRA DI SCRITTURA, in cui verranno disposti i caratteri digitati da tastiera.
Le modalità di definizione di questo, sono analoghe a quelle descritte nel paragrafo BOX (vedi). Dopo aver definito la FINESTRA DI SCRITTURA apparirà al suo interno un cursore che ci segnala la disponibilità di GEOPAINT ad accettare da tastiera il testo desiderato.
Il tasto RETURN in questo caso non serve per terminare ogni parola composta, ma deve essere usato come un "a capo". Dopo aver digitato qualche parola, si potrà scegliere lo stile più adatto al documento utilizzando le vaie opzioni disponibili nella FINESTRA DI STATO. Fatto ciò si dovrà digitare l'intero testo desiderato, correggendo col tasto INST/DEL gli eventuali errori commessi. Nel caso in cui l'intero testo non possa essere contenuto nella FINESTRA DI SCRITTURA è possibile ridefinirne un'altra secondo le solite modalità. Ciò annulla quanto è stato scritto con lo strumento TRASFERIBILI ma salva il testo composto che riappare nella nuova FINESTRA DI SCRITTURA.

RETTANGOLI VUOTI

Questo strumento, indicato con 16 in fig. 2.2, vi consente di tracciare rapidamente qualsiasi rettangolo. Le modalità d'uso sono quelle consuete che vengono utilizzate nella definizione della BOX (vedi).

RETTANGOLI PIENI

Talvolta è utile disporre di arre rettangolari colorate. Questo risultato può essere facilmente raggiunto utilizzando in combinazione RETTANGOLI VUOTI e RUBINETTO oppure, più rapidamente, lo strumento RETTANGOLI PIENI (punto 17 in fig. 2.2). Il suo uso è simile al precedente (vedi).

COMPASSO

Qualsiasi circonferenza può essere rapidamente tracciata utilizzando lo strumento che in fig. 2.2 è indicato con 18.
L'uso di COMPASSO si articola in due passi: a) ci si posiziona sul centro della circonferrenza da tracciare e si preme il tasto fire. b) ci si spostra lungo un raggio fino a raggiungerne l'estremità e quindi si preme nuovamente il tasto di fire.
Durante questa operazione si osserverà una circonferenza che si modifica dinamicamente in modo da avere come raggio il segmento compreso fra il centro precedentemente definito e la posizione attuale della freccia.

CERCHIO

Questo strumento (fig. 2.2 punto 19) permette di tracciare rapidamente un qualsiasi cerchio. Il suo uso è identico a quello di COMPASSO.

UNDO

Ogni operazione compiuta con gli strumenti sopra descritti non è immediatamente riportata in maniera definitiva sul foglio da disegno. Ogni cambiamento diviene definitivo solo quando se ne intraprende un'altro.
Se prima di apportare una nuova modifica si desidera annullare l'ultima operazione compiuta ci si può utilmente servire dello strumento UNDO. Per utilizzarlo sarà sufficiente attivarlo.

IL COLORE ATTUALE

In fig. 2.2 è indicato con 21 un quadrato riempito con il colore che abbiamo correntemente selezionato. In effetti non si tratta propriamente di un colore

ma di un esempio del tratteggio che si otterrebbe utilizzando lo strumento RUBINETTO.
Per modificare il colore attuale è necessario che ci venga mostrata l'intera tavolozza dei colori disponibili da cui dovremo scegliere. Ciò si ottiene posizionando la freccia all'interno del quadrato più sopra menzionato e premendo il tasto di fire.
Fatto ciò si sceglierà mediante il joystick il colore desiderato.

2.4 I PULL-DOWN MENU

Ciascuno di questi può essere attivato posizionadosi all'interno del rettangolo che contiene il nome che lo individua e premendo il tasto di fire. Vengono così presentate le varie opzioni disponibili. Se fra queste troviamo quella che ci interessa sarà sufficiente posizionarvisi e premere nuovamente il tasto di fire. Diversamente il pull-down menu può essere disattivato semplicemente uscendo dall'area occupata dalle varie opzioni.

FONTS

Come già accennato, GEOPAINT mette a disposizione dell'utente diverse serie di caratteri detti FONTS.
La particolare serie di caratteri desiderata può essere utilizzata selezionandola dal pull-dawn menu FONTS. La scelta di una di queste provoca l'automatica attivazione dello strumento TRASFERIBILI. Quando viene scelto un particolare FONT ci viene anche richiesto di selezionare la dimensione desiderata per ogni carattere della serie. La selezione della dimensione del carattere viene effettuata, come di consueto, premendo il tasto di fire sulla dimensione desiderata.

OPTIONS

PIXEL EDIT Questa opzione consente di effettuare con grade precisione ritocchi su una particolare area del grafico che correntemente abbiamo sul video. Infatti, poichè il disegno su cui operiamo è realizzato accostando una serie di pixel, ossia di puntini, l'opzione PIXEL EDIT ci consente di accedere con facilità a ciascuno di essi; potremo così lavorare su un'immagine ingrandita di una porzione del disegno contenuto nell'AREA GRAFICA.

L'uso di questo strumento è semplice: dopo averlo attivato viene mostrato sul video un rettangolo, detto ZOOM AREA, che si sposterà seguendo i movimenti del joystick. Poichè il disegno ivi contenuto è quello che verrà ingrandito, dovremo posizionare ZOOM AREA sul particolare che ci interessa e successivamente premere il tasto di fire per comunicare a GEOPAINT che quella è l'area selezionata. Subito ci verrà mostrata l'immagine ingrandita. Su questa ora possiamo lavorare con alcuni dei consueti strumenti che GEOPAINT mette a disposizione.
Infatti in modalità PIXEL EDIT non sono disponibile tutti gli attrezzi da disegno.

NORMAL EDIT Questa opzione è quella che ci permette di di uscire dalla modalità PIXEL EDIT. Per attivarla è sufficiente aprire il pull-down menu e selezionarla.

CHANGE BRUSH Lo strumento pennello, come già accennato permette di realizzare diversi tipi di tratti. Quest'opzione serve a selezionare quello più adatto fra i vari disponibili.
Quando CHANGE BRUSH è attivato, secondo le solite modalità, vengono presentati nella finestra di stato i tratti fra cui possiamo scegliere. Quello correntemente selezionato viene evidenziato in un quadrato. Per sostituirlo sarà sufficiente posizionarsi sul tratto desiderato e premere il tasto di fire.

DISPLAY PAGE Poichè sul video ci viene mostrata solo l'AREA GRAFICA che contiene una piccola parte di quanto abbiamo tracciato sul nostro foglio da disegno, facilmente si può avere un sensazione di smarrimento dovuta alla mancanza di una visione di insieme.
DISPLAY PAGE soddisfa a questa esigenza permettendo di rappresentare sul video l'intero documento su cui stiamo lavorando. Per utilizzare questa opzione sarà sufficiente attivarla.
Verrà mostrato l'intero foglio e la scritta OK su cui dovremo posizionarci e premere il tasto di fire per poter continuare a disegnare.

UPDATE FILE Per comprendere il significato di questa opzione e della seguente, è necessario sapere che GEOPAINT tiene nella memoria centrale del calcolatore solo il contenuto dell'AREA GRAFICA. Sul dischetto invece è disponibile l'intero foglio da disegno. UPDATE FILE ci permette di ricopiare su questo le modifiche che abbiamo apportato sul video. Tuttavia è bene precisare che GEOPAINT in certi casi effettua automaticamente questa operazione. Un caso tipico è il movimento dell'AREA GRAFICA mediante l'apposito strumento.

RECOVER FROM FILE L'effetto di questa opzione è opposto a quello della precedente. Infatti, nel caso che siano stati combinati pasticci nell'AREA GRAFICA, RECOVER FROM FILE permette di ripristinare quanto avevamo precedentemente a diposizione su dischetto. Le modalità d'uso si limitano all'attivazione di questa opzione.

EDIT

CUT Questa opzione, come le due seguenti, deve essere usata in combinazione con BOX che definisce un rettangolo all'interno dell'AREA GRAFICA. Infatti CUT toglie dal documento il contenuto di questo e lo pone in un'area di memoria presente su disco detta PHOTO SCRAP. Un nuovo impiegho di CUT provocherà la perdita di del precedente contenuto di PHOTO SCRAP. Pertanto per togliere un particolare dal nostro documento dovremo: a) definire mediante BOX il particolare che intendiamo eliminare. b) attivare l'opzione CUT.

COPY Questa opzione è del tutto simile alla precedente. Vi si differenzia poichè non toglie dal disegno il contenuto di BOX ma lo copia in PHOTO SCRAP.

PASTE A nulla serve depositare o copiare in PHOTO SCRAP particolari di un documento se questi non possono venir riutilizzati altrove. PASTE provvede a copiare il contenuto di PHOTO SCRAP nel BOX che dovremo aver definito prima di attivare questa opzione.

FILE

CLOSE Al termine di una sessione di lavoro (cioè quando abbiamo finito un disegno o comunque quando intendiamo interromperlo) dobbiamo attivare quest'opzione. In risposta ci verrà richiesto, mediante un menu identico a quello che si incontra appena si apre GEOPAINT da DESKTOP, quali azioni intendiamo intraprendere. Possiamo creare un nuovo documento, aprirne uno già preesistente od abbandonare il tavolo da disegno ritornando così a DESKTOP. La scelta fra queste alternative va eseguita mediante il joystick.

PRINT Selezionando questa opzione si manda in stampa l'intero documento corrente. Poichè GEOS supporta diversi tipi di stampanti bisognerà accertarsi che il primo PRINTER DIVER che si incontra sul taccuino direttorio sia quello compatibile con la stampante che si intende utilizzare. Per un ulteriore approfondimento si rimanda all'esame dell'accessorio Choose Printer visibile da DESKTOP.

RENAME Talvolta può essere utile cambiare il nome del documento corrente. RENAME permette di soddisfare a questa necessità. Appena si attiva quest'opzione ci viene richiesto il nuovo nome. Questo, digitato da tastiera e terminato da RETURN, non può superare i 16 caratteri. L'opzione CANCEL che è attivabile mentre si digita il nuovo nome serve ad interrompere l'operazione di RENAME senza che il nome del documento venga cambiato.

QUIT Questa opzione agisce esattamente come la CLOSE ma, senza chiederci alcunchè, ci rimanda a desktop.

GEOS

Per l'esame di quasi tutte le opzioni che sono selezionabili da questo pull-down menu si rimada al capitolo che tratta gli accessosi di GEOS. Qui trattiamo brevemente solo l'opzione GEOPAINT INFO.
Questa, una volta attivata, mostra alcune informazioni riguardanti il programma GEOPAINT.

CAPITOLO 3

L'USO DI GEOWRITE, IL WORDPROCESSOR

3.1 INTRODUZIONE

In questo capitolo tratteremo l'uso della seconda applicazione contenuta sul vostro dischetto e cioè il wordprocessor. Questo è un programma che vi permette di scrivere testi in vari stili e formati, di memorizzarli su dischetto e infine di stamparli. Per cominciare tornate al Desktop e visualizzate la prima pagina del taccuino del direttorio.
Aprite dunque l'applicazione GEOWRITE con i metodi già visti. Dopo alcuni istanti il GEOS vi presenterà tre possibili scelte:
CREATE vi permette di creare un nuovo documento

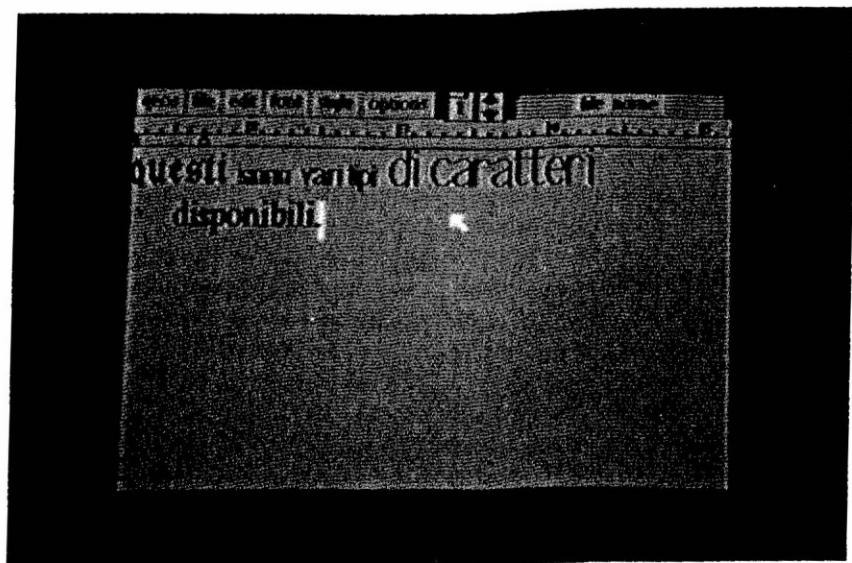



Figura 3.1 - GEOWRITE, il word processor.

OPEN vi permette di accedere a un documento creato in precedenza

QUIT vi permette di tornare al Desktop

Potete quindi cominciare a creare il vostro primo documento selezionando l'opzione CREATE.
Il GEOS procederà quindi a chiedervi il nome che volete assegnare al testo: sceglietene uno qualunque e inseritelo non dimenticando di schiacciare il tasto di RETURN alla fine.
A questo punto apparirà sul vostro schermo un foglio in bianco su cui potrete scrivere ciò che vorrete (fig. 3.1). In realtà sul video vi è rappresentata solo una piccola parte della pagina che avete a disposizione: questa è più larga e più lunga. Per avere un'idea della situazione osservate il riquadro in alto al centro (fig. 3.2): rappresenta l'intera pagina a disposizione.
Il rettangolino più scuro all'interno indica la zona visualizzata. Questo può essere spostato a piacere: - posizionate la freccia sul rettangolino e premete il tasto di fire - a questo punto potrete trascinarlo muovendo la manopola del joystick, e depositarlo sulla zona da visualizzare premendo nuovamente il tasto di fire. Il numero all'interno del riquadro indica la pagina su cui siete posizionati.

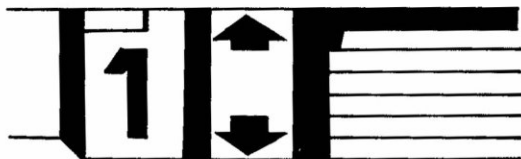



Figura 3.2 - Particolare della pagina di lavoro.

Un'altro modo per spostare la zona rappresentata all'interno di una pagina, consiste nel posizionare la freccia su uno dei due triangolini situati a destra del riquadro e di azionare il tasto di fire: potrete così avanzare o indietreggiare di una linea.
Sulla riga inferiore si trova una scala graduata da 1 a 7 che, oltre ad essere utile per fissare margini e tabulatori, vi consente di stabilire la posizione orizzontale di un qualsiasi carattere all'interno di una linea di testo. Provate quindi a scrivere una riga molto lunga (senza andare a capo): se all'inizio veniva visualizzata la zona dal 1 al 5, a un certo punto vi verrà mostrata la seconda parte della pagina dal 4 al 7 (sono numeri leggibili sulla scala graduata). Per tornare a visualizzare la parte sinistra potete:

— andare a capo, premendo RETURN
— utilizzare il riquadro visto prima spostando il rettangolino nella posizione voluta
— muovere mediante il joystick la freccia, in modo da cercare di superare il limite sinistro della zona visualizzata; vedrete così apparire la prima parte della vostra riga.

Gli ultimi due metodi sono applicabili anche per effettuare l'operazione inversa. Per il terzo bisogna però dirigere la freccia verso destra. Questi cambiamenti dello schermo possono disturbare chi scrive; percui si usa fissare un margine destro sulla tacca 5. Al momento della stampa, lo sposterete secondo le vostre esigenze.

Per fissare il margine destro procedete così:

— spostatevi sulla seconda metà della pagina in modo da visualizzare la tacca 7
— sotto di questa si trova un indice che segnala la posizione del margine
— mediante la manopola del joystick, posizionate la freccia sopra tale indice e premete il tasto di fire - ora potrete spostare il margine orizzontalmente
— fissate il margine sulla tacca desiderata premendo il tasto di fire.

Per spostare il margine sinistro procedete nello stesso modo, spostando però l'indice situato sotto la tacca 1. I margini definitivi vengono fissati una volta scritto e corretto il testo e dipendono dal formato della carta su cui scrivere. Il formato massimo è quello A4 verticale (modulo continuo 234 mm per 12", 72 linee per foglio).
In ogni caso ricordatevi che il GEOS provvede già a lasciare dello spazio libero dai limiti destro, sinistro, inferiore e superiore della pagina. Dimensioni diverse richiedono uno spostamento dei margini e l'inserimento di cambi di pagina (page breaks vedi il capitolo seguente) in modo che la stampa non esca dal foglio. Noi vi consigliamo sempre di fare dei tentativi in modo da ottenere la stampa migliore.

Ulteriori particolarità in fase di scrittura Il cursore è rappresentato da una linea verticale lampeggiante. Ogni tasto premuto genera un carattere che viene inserito nel testo nel luogo dove si trova il cursore; questo viene fatto avanzare di un posto.
Il fatto che voi inseriate caratteri significa che non cancellerete mai nulla sul vostro testo semplicemente scrivendo delle lettere, come avveniva quando scrivevate programmi in BASIC. Questa è una caratteristica molto utile comune a tutti i wordprocessors e imparerete ad apprezzarla con il tempo.

Per cancellare dei caratteri non vi rimane che usare il tasto di DELETE (DEL in alto a destra sulla tastiera) che eliminirà quello situato a sinistra del cursore; diversamente si può attivare un pezzo di testo e toglierlo seguendo le opzioni che vedremo però in seguito.
Un'altra caratteristica interessante di GEOWRITE è quella del wordwrap; questa fa in modo che i margini non interrompano a metà le parole che voi avete scritto. Infatti nel caso in cui la parola non possa essere contenuta sulla riga corrente, viene presa e spostata su quella seguente.
Inoltre come avrete notato il cambio di linea risulta essere automatico. Per imporne uno e sufficiente premere il tasto RETURN; è importante notare che questo è visto come un normale carattere ed è quindi eliminabile usando il tasto di DELETE. Provate ad esempio a scrivere la frase seguente: "Questo è un esempio di come" (RETURN) "ricongiungere due linee"
Ora prendete in mano in joystick e puntate la freccia sulla prima "r" di "ricongiungere": premete il tasto di fire e vedrete che il cursore si sarà spostato proprio nel punto indicato dalla freccia. Se ora provate a inserire degli spazi vedrete che questi sposteranno la linea inferiore verso destra; ora premete il tasto DELETE in modo da far tornare la parola "ricongiungere" contro il margine sinistro.
A questo punto premendo ancora una volta il tasto DELETE ottenerrete l'eliminazione del carattere RETURN e le due linee si ricongiungeranno. In pratica tale carattere è situato immaginariamente subito dopo l'ultimo della linea precedente: ben si comprende dunque il significato dell'operazione vista sopra.

Inoltre avete potuto vedere come sia possibile posizionare il cursore mediante il joystick. A tale proposito provate a scrivere le linee: "Gli spazi qui a destra" (RETURN) "non sono veri caratteri, bensì costituiscono una zona vuota" Cercate ora di posizionare il cursore sulla prima riga, nella zona senza caratteri a destra della parola "destra".
Non ci riuscirete: esso infatti si piazzerà sempre subito dopo la "a". La ragione di tale comportamento è che fra "destra" e "non" c'è un solo carattere e cioè RETURN: gli spazi rimasti sulla destra NON sono caratteri ma zone vuote.
Per cui ogni volta che cercherete di posizionare il cursore in una tale regione, ricordatevi che esso si metterà sempre subito dopo il primo carattere che incontrerà viaggiando immaginariamente verso sinistra (e verso l'alto).
Quanto visto fin qui ci aiuta a capire come i wordprocessors memorizzino i testi. Questi sono costituiti da una lunga sequenza di caratteri: alcuni alfanumerici (1,2,3,...,a,b,c,.....) e altri di controllo (RETURN per esempio). Al momento di scrivere il testo su schermo, il programma stampa tutti i caratteri in sequenza, facendo attenzione a rispettare i margini imposti: in definitiva

questi non modificano la struttura del file memorizzato.
Se scrivete molte linee, arriverete a un punto in cui la pagina verrà cambiata automaticamente. Un tale cambio può essere imposto selezionando l'opzione "page break" dal menu "options". In tal modo il wordprocessor inserisce nel testo un carattere di controllo che impone il cambio della pagina. Questo può essere cancellato con le stesse modalità viste per RETURN, riunendo così due sezioni di testo poste su due fogli separati ma contigui. Supponiamo allora di essere in pagina 1:

— selezionate dal menu superiore l'opzione "page break"
— dopo alcuni secondi appare la pagina 2 (vuota);
— scrivete alcuni caratteri (qualunque)
— posizionate il cursore sul primo carattere della pagina 2, in modo che sia in alto a sinistra attaccato al margine
— premete il tasto di DELETE
— dopo alcuni secondi apparirà una finestra supplementare che vi chiederà se cancellare l'ultimo carattere della pagina precedente, in questo caso quello di controllo "page break": voi selezionate la scelta O.K.
— in tal modo avrete riattaccato la pagina 2 alla 1

Bisogna infine dire che "page break" inserisce un foglio nuovo nel testo, incrementando di uno la numerazione degli eventuali fogli seguenti. Per chiudere il testo di prova che abbiamo creato e tornare al Desktop, selezionate l'opzione "quit" dal menu superiore "file"; sfogliando il taccuino del direttorio troverete una nuova icona, quella del testo che avete appena scritto. Per tornare a modificarlo, è sufficiente effettuarvi un doppio click sopra.
Per cancellarlo, trascinatelo sul contenitore delle immondizie.

3.3 FONT: come cambiare caratteri

Esaminiamo ora una delle caratteristiche più interessanti di GEOWRITE, vale a dire la possibilità di utilizzare più tipi di caratteri, di diversa dimensione nello stesso testo. Per accedere a tale possibilità bisogna attivare il menu "font". Comparirà una lista di tipi di caratteri (font) disponibili; questi dipendono dai file font memorizzati sul dischetto in uso (sul taccuino del direttorio sono rappresentati da icone contenenti la scritta "FONT"). Noterete che un asterisco indica il tipo di stampa attualmente in uso. Per sceglierne uno nuovo selezionate il nome che vi interessa; successivamente apparirà un altro menu da cui dovrete scegliere l'altezza in numero di punti del carattere. Per modificare solo quest'ultima è sufficiente, dopo avere attivato il medesi-

mo menu, selezionare il font in uso, e in seguito modificarne le dimensioni a piacere. La scelta della spaziatura fra caratteri e fra linee successive è automatica, e fa sempre sì che non vi siano sovrapposizioni.

3.4 Gli stili di scrittura

Una volta scelto il font più adatto, può rendersi necessario in certi punti del testo un cambiamento dello stile di scrittura; per fare ciò attivate il menu "style". L'asterisco indica quello attualmente in uso; se si trova a sinistra di "plain text" vuol dire che la scrittura è normale. Selezionate lo stile voluto nel solito modo; la differenza con gli altri menu superiori, consiste nel fatto che più stili possono essere contemporaneamente attivi. Potrete così scrivere,per esempio, in grassetto e sottolineato (bold e underlined). Per eliminarne uno, attivate il menu superiore "style", puntate la freccia sul nome relativo e premete il pulsante di fire. Riattivando il menu noterete la sparizione dell'asterisco dal fianco dello stile disabilitato.
Ogni volta che posizionate il cursore mediante il joystick in una zona di testo, il font e lo stile con cui scriverete vengono determinati dal carattere precedente. Per cambiare bisogna quindi attivare i menu "font" e "style" e scegliere le opzioni che interessano.

3.5 Attivazione di zone di testo

Un particolare uso del joystick e del cursore, riguarda l'attivazione di sezioni del testo. Per compiere tale operazione è sufficiente: - posizionare il cursore sul primo carattere da attivare - premere il tasto di fire e non rilasciarlo - muovere la manopola del joystick in modo da attivare una parte del testo visualizzato; potete muovervi in tutte le direzioni ma non dimenticate di tenere schicciato il pulsante di fire - rilasciare il tasto di fire quando avrete attivato la zona voluta La parte di testo così attivata viene messa in evidenza tramite lo scambio del colore di fondo con quello dei caratteri (reverse).
A questo punto le operazioni possibili sono le seguenti.
Cancellazione: È sufficiente premere il tasto di DELETE (INST/DEL) Sostituzione: Se dovete sostituire la sezione attivata con un testo diverso, scrivetelo subito: GEOWRITE provvederà a cancellare il testo vecchio sostituendolo con quello nuovo Cambiamento dei font e dello stile: Potete cambiare il tipo e lo stile di tutti i caratteri attivati, selezionando le opzioni desiderate dai menu superiori "font" e "style" (vedi paragrafi prcedenti) Trasferimento in text

scrap: Nella sezione dedicata a GEOPAINT abbiamo visto l'uso di photo scrap; un file molto simile esiste pure per i testi e si chiama text scrap. Per trasferirvi il blocco che è stato attivato è sufficiente selezionare dal menu "edit" le opzioni:

— CUT , se si intende togliere il blocco per trasferirlo altrove nel testo
— COPY , se si vuole copiarlo Una volta copiato in text scrap, il testo può essere incollato ovunque nel nostro documento.

È sufficiente posizionare il cursore nel punto dove dovrà essere inserito e selezionare dal menu "edit" l'opzione "paste". A questo punto comparirà un altro menu che vi chiederà se incollare delle immagini (GEOPAINT) o del testo: selezionate quest'ultimo (opzione "text").

Questa operazione di inserimento non svuota il contenuto di text scrap e quindi può essere ripetuta più volte. Da notare che i caratteri vengono trasferiti senza essere modificati. Se aveste in precedenza scelto l'opzione "picture", avreste copiato nel vostro documento l'immagine contenuta in photo scrap. In tal modo è possibile combinare testi e disegni, per documenti estremamente professionali. Siccome text scrap può contenere un blocco di testo alla volta, si è sentita la necessità di creare un raccoglitore di tali parti di testo: il suo nome è text album e il suo uso sarà approfondito nella sezione dedicata agli accessori del GEOS.

3.6 Visualizzazione delle pagine

Una volta creato un documento a più fogli diventa utile il poter passare semplicemente da uno all'altro. Per fare ciò è sufficiente attivare il menu "options" e selezionare l'opzione voluta. LAST PAGE: visualizza la pagina precedente NEXT PAGE: visualizza la pagina seguente GOTO PAGE: il GEOS vi chiederà il numero della pagina da visualizzare; inseritelo non dimenticando di premere RETURN alla fine HIDE PICTURES : se avete inserito immagini, questa opzione non le farà apparire, aumentando la velocità di visualizzazione SHOW PICTURES : le immagini vengono di nuvo mostrate PAGE BREAK : inserisce un cambio pagina

3.7 Gestione dei file

Avrete sicuramente notato che mentre scrivete un testo il disco non rimane sempre inattivo e anzi risulta essere molto spesso utilizzato. Questo succede in

quanto GEOPAINT tiene nella memoria centrale solo la parte visualizzata, e ogni volta che questa cambia, il file viene aggiornato. Ciò può essere esplicitamente richiesto selezionando "UPDATE FILE" dal menu "file". In tal modo sarete sicuri che anche le ultime modifiche sono state salvate su disco, e potrete spegnere il COMMODORE 64 senza preoccupazioni.
Esiste poi l'opzione inversa, che vi permette di ripristinare la zona visualizzata in caso che abbiate fatto delle modifiche non volute; questa è "recover from file" sempre presente nel menu "file". Da notare che tale operazione ha senso se e solo se il contenuto del file non è stato aggiornato già automaticamente da GEOWRITE (basta osservare il drive: ogni volta che funziona viene operata una modifica dei testi su disco).
Nel menu "file" sono poi disponibili altre opzioni: CLOSE : aggiorna il documento con le ultime modifiche e va al menu iniziale di GEOWRITE PRINT : stampa il documento PREVIEW PAGE : visualizza l'aspetto finale della pagina su cui si sta lavorando; per tornare al testo posizionatevi su O.K. e premete il tasto di fire QUIT : aggiorna il documento con le ultime modifiche e va al Desktop

3.8 Uso dei tabulatori

L'uso dei tabulatori permette di raggiungere una predefinita posizione orizzontale all'interno del testo, premendo contemporaneamente i tasti "CTRL" e "I". Il tabulatore definisce tale punto; per piazzarne uno, seguite la procedura seguente. - usando il joystick posizionate la freccia fra la scala graduata e il riquadro del testo nel punto desiderato - premete il tasto di fire e vedrete apparire un indicatore che segnala la presenza del tabulatore.
Per eliminare un tabulatore, puntate la freccia sopra l'indicatore relativo, e dopo aver premuto il tasto di fire, trascinatelo sul margine sinistro; qui rilasciatelo premendo ancora il pulsante di fire.

CAPITOLO 4

GLI ACCESSORI DI GEOS

I programmi che sono disponibili all'interno di GEOS appartengono a due classi: le APPLICAZIONI e gli ACCESSORI.
Questi ultimi si differenzano dalle APPLICAZIONI perchè sono utilizzabili in qualsiasi momento. Ciò vuol dire che è possibile utilizzare l'accessorio CALCULATOR anche se abbiamo attivata (cioè stiamo eseguendo) l'applicazione GEOPAINT.
Non possiamo invece utilizzare GEOWRITER durante l'esecuzione di GEOPAINT perchè sono entrambe delle applicazioni.
Per utilizzare un qualsiasi accessorio è sufficiente selezionarlo dal pull-down menu GEOS che è sempre presente sullo schermo.
Se proviamo ad attivare da DESKTOP il pull-down menu potrete scegliere fra i seguenti accessori:
GEOS INFO
DESKTOP INFO
CHOSE PRINTER
PREFERENCE MGR
NOTE PAD
PHOTO MANAGER
TEXT MANAGER
ALARM CLOCK
CALCULATOR
In realtà queste non sono tutti degli accessori veri e propri. Infatti GEOS INFO e DESKTOP INFO, se attivati, presentano su video delle informazioni sul sistema operativo GEOS e su DESKTOP.
Per disattivarli è sufficiente premere il tasto di fire. Anche CHOOSE PRINTER non può essere considerato un accessorio vero e proprio poichè serve solo a mostrare il DRIVER DI STAMPA correntemente selezionato. Questo è un programma che vi permette di stampare i vostri documenti, e dipende dal tipo di apparecchio usato.
Sul vostro dischetto ce ne sono cinque rappresentati con icone che mostrano una stampante. Viene automaticamente selezionato il primo DRIVER incontrato sul taccuino del direttorio; dovrete quindi spostare quello relativo alla vostra stampante, in testa agli altri. Esaminiamo ora uno ad uno i veri accessori di GEOS:

NOTE PAD

Questo accessorio costituisce un taccuino sempre disponibile per registarvi annotazioni di ogni genere. Per aprirlo è sufficiente attivare dal pull-down menu l'opzione corrispondente. Vi si presenterà dopo qualche istante una schermata simile a quella riportata in fig. 4.2.

L'uso di NOTE PAD è semplicissimo: per inserire un appunto è sufficiente digitarlo da tasiera mentere per correggerlo o toglierlo si ricorre al tasto INST DEL. Per voltare le pagine ci si posiziona sull'angolo in basso a sinistra del taccuino e si preme il tasto di fire. Analogamente, per chiudere NOTE PAD, ci si posiziona sul tasto posto in alto a destra e si preme il tasto di fire.

PHOTO MANAGER

In GEOPAINT, quando si è illustrato l'uso del pull-down menu EDIT, si è parlato di un file di sistema detto PHOTO SCRAP. Su questo si poteva memorizzare un'immagine. Ciò risultava di grande utilità poichè consentiva di muovere, copiare e cancellare dei particolari di un disegno. Tuttavia in PHOTO SCRAP non era possibile salvare più di un'immagine per volta e ciò impediva di realizzare un'archivio grafico.

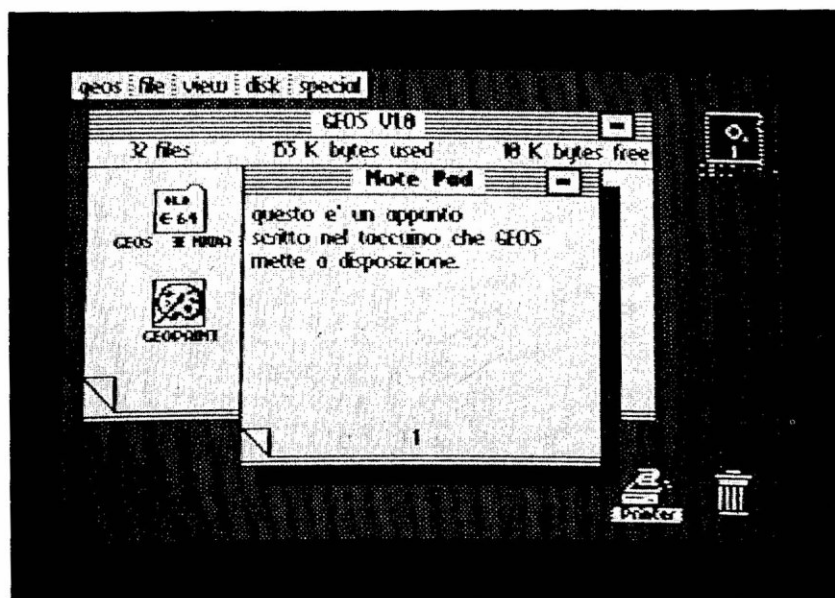



Figura 4.2 - L'accessorio NOTE PAD.

Con PHOTO MANAGER questo problema è superato. Infatti questo accessorio è un vero e proprio gestore dell'archivio costituito da PHOTO ALBUM. Le tre operazioni che PHOTO MANAGER consentedi compiere sono:

1) AGGIUNTA DEL DISEGNO CONTENUTO IN PHOTO SCRAP AL PHOTO ALBUM
2) COPIARE UN DISEGNO DAL PHOTO ALBUM AL PHOTO SCRAP
3) TOGLIERE UN DISEGNO DAL PHOTO ALBUM TRASFERENDOLO IN PHOTO SCRAP

Veniamo ora al dettaglio di queste due operazioni. Dopo aver aperto PHOTO MANAGER vi si presenterà una videata simile a quella di fig. 4.3. Per copiare il contenuto di PHOTO SCRAP in PHOTO ALBUM sarà sufficiente selezionare dal pull-down menu EDIT l'opzione PASTE. Diversamente se intendiamo copiare su PHOTO SCRAP l'immagine che PHOTO MANAGER ci sta mostrando sul video si dovrà attivare dal pull-down menu EDIT o l'opzione CUT oppure l'opzione COPY.

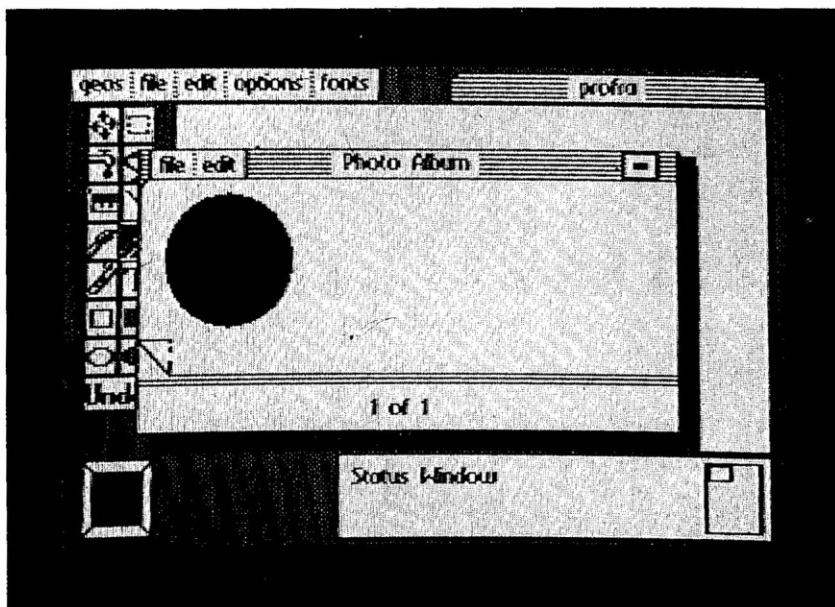



Figura 4.3 - L'accessorio PHOTO ALBUM.

A differenza di COPY, CUT provvede anche a togliere da PHOTO ALBUM l'immagine che viene copiata in PHOTO SCRAP. Se desideriamo sfogliare PHOTO ALBUM dovremo clickare sull'angolo in basso a sinistra del riquadro che PHOTO MANAGER utilizza per presentarci le immagini di PHOTO ALBUM.

Per chiudere questa applicazione potremo indifferentemente posizionarci sul tasto posto in alto a destra nella finestra in cui PHOTO MANAGER opera e premere il tasto di fire oppure selezionare dal pull-down menu FILE l'opzione CLOSE.

TEXT MANAGER
Il modo di operare di questo accessorio è quasi identico a quello di PHOTO MANAGER e pertanto indicheremo soltato le differenze che vi si possono riscontrare. Innanzi tutto i file su cui TEXT MANAGER lavora sono TEXT ALBUM e TEXT SCRAP.
Questi, pur svolgendo ruoli assolutamente analoghi a quelli di PHOTO ALBUM e PHOTO SCRAP, se ne differenziano perchè non sono adatti a trattare disegni ma soltanto testi, intesi secondo la definizione più sopra riportata. La seconda differenza fra queste due applicazioni è costituita dalla schermata che si presenta quando si apre TEXT MANAGER. Questa, pur essendo quasi identica a quella usata da PHOTO MANAGER, offre in più la possibilità di scorrere il testo utilizzando le due frecce verticali poste sul margine inferiore del riquadro in cui TEXT MANAGER mostra TEXT ALBUM.

ALARM CLOCK
Questo simpatico accessorio funziona in maniera simile ad un orologio da polso che ci mostra l'ora e che dispone di una sveglia. Questa una volta predisposta suonerà anche se ALARM CLOCK non è aperto. In fig. 4.4 è stata riportata la schermata che si ottiene selezionando questo accessorio dal pull-down menu GEOS.
Vi sono indicati con:

1) il visore da cui si legge l'ora e su cui si imposta la sveglia.
2) il pulsante/spia che permette di selezionare la modalità di funzionamento OROLOGIO o SVEGLIA, l'una indicata con il simbolo di un orologino l'altra con una CAMPANELLA.
3) il pulsante che attiva o disattiva la sveglia. Questo tasto agisce agisce solo in modalità sveglia.
4) il tasto che ci permette di chiudere questa applicazione.
5) la spia che ci avverte dell'inserimento della sveglia.

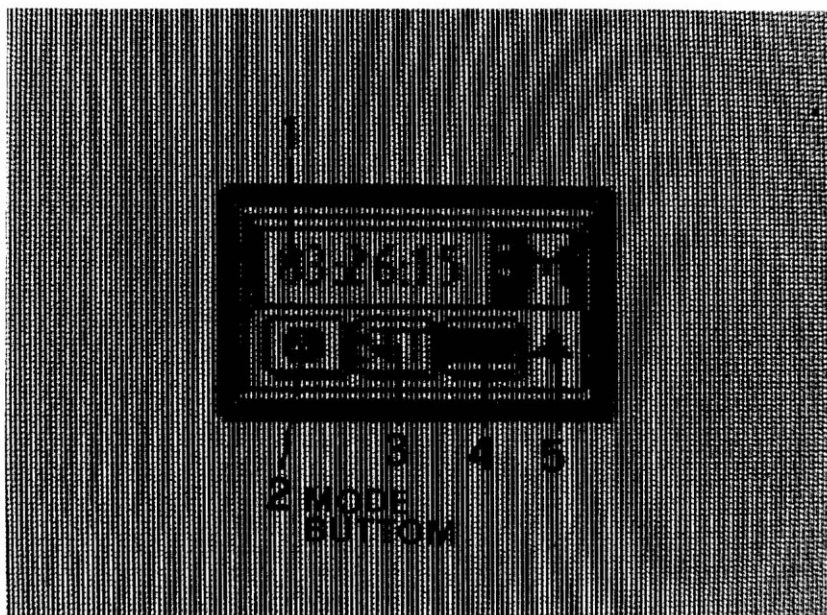



Figura 4.4 - L'accessorio ALARM CLOCK.

Desiderando impostare l'ora bisogna procedere così:
— accertarsi di essere in MODALITÀ OROLOGIO, osservando il simbolo mostrato dal PULSANTE/SPIA; se non foste in tale modalità, posizionatevi sopra di questo e premete il tasto di fire.
— digitare il tasto A oppure P se si intende inserire un'ora antimeridiana oppure pomeridiana
— digitare ora, minuti e secondi
— posizionatevi sul pulsante di set e premere il tasto fire.

Desiderando invece impostare l'ora di sveglia si dovranno eseguire queste altre operazioni:
— accertarsi di essere in modalità SVEGLIA, osservando il simbolo mostrato dal PULSANTE/SPIA; se non foste in tale modalità, posizionatevi sopra di questo e premete il tasto di fire.
— digitare il tasto A oppure P se si intende inserire un'ora antimeridiana oppure pomeridiana
— digitare ora, minuti e secondi

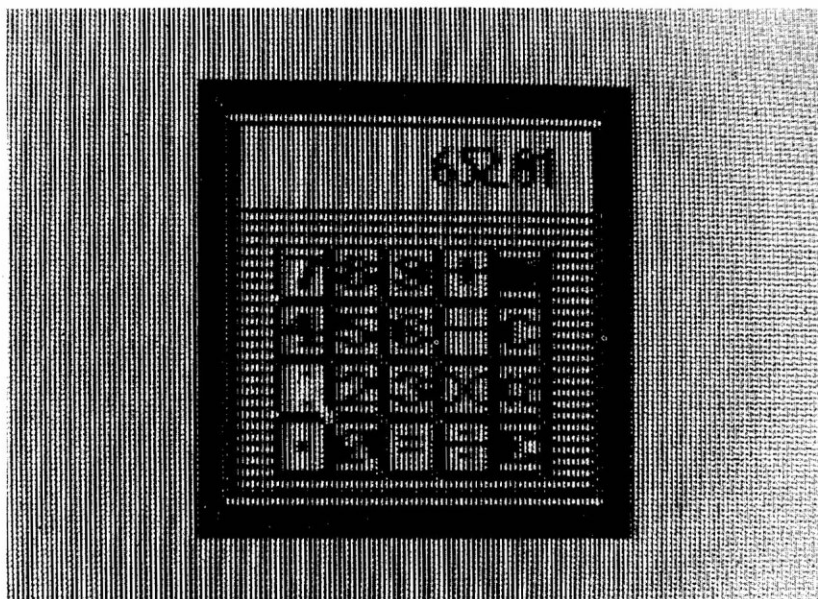



Figura 4.5 - L'accessorio CALCULATOR.

— posizionatevi sul pulsante di set oppure sul pulsante mode e premete il tasto di fire.

CALCULATOR

Anche questo accessorio è di uso semplicissimo. Una volta aperto vi si presenterà una schermata analoga a quella riportata in fig. 4.5. I tasti sono quelli consueti di una normale calcolatrice da tasca e sono azionabili posizionandovisi sopra e premendo il fire. Il tasto di OFF, posto in alto a destra, serve per chiudere questo accessorio. Anche nell'uso non vi sono differenze rispetto alle ormai diffuse calcolatrici tascabili.

CAPITOLO 5

ALCUNI CONSIGLI FINALI

5.1 CREAZIONE DI UN WORKDISK

Siccome il vostro dischetto GEOS risulta essere quasi interamente occupato, diventa necessario creare un disco di lavoro su cui siano presenti solo i file indispensabili. A tale proposito tratteremo l'esempio di un workdisk dedicato al wordprocessor.

— Formattate un disco con il GEOS; il nome può essere qualunque
— Copiatevi sopra dal disco sistema i file che ritenete necessari

Desktop: non è fondamentale, ma è più comodo averlo su ogni disco
Geowrite: è l'applicazione cui è dedicato questo disco di lavoro
Font:copiate i file di font che ritenete necessari; potrete sempre comunque aggiungerne uno
Stampante: copiate un solo file (Printer driver), quello relativo alla vostra stampante (per es. Commodore)
Text manager: copiatelo solo se volete usare il text album; quest'ultimo può essere copiato da altri dischetti, ma su ognuno non ne possono coesistere due
Photo Manager: copiatelo insieme al photo album, se e solo se intendete incollare immagini nel vostro testo

Ricordate che meno file copiate più spazio avrete a disposizione per i vostri documenti. Questo disco NON può essere usato per caricare il sistema GEOS: per compiere tale operazione dovrete usare sempre il disco di sistema.
A volte può essere noioso il dover sfogliare le pagine del taccuino del direttorio per aprire il documento su cui si sta lavorando. Allora è utile applicare un piccolo trucco, consistente nel trascinare l'icona del file sul fondo, rilasciandola sotto il taccuino, in modo che sia sempre visibile indipendentemente dal foglio visualizzato. Il GEOS si ricorderà di tale operazione e ogni volta che aprirete il disco di lavoro vedrete comparire, in basso e separato, il file. In tale zona avete posto per quattro icone. Per aprire tale file sarà sufficiente effettuare il solito doppio click; potrete poi riportarlo nel taccuino quando vorrete.

5.2 Come legare i vostri programmi Basic al GEOS

Abbiamo visto in un capitolo precedente come fosse possibile lanciare i vostri programmi basic dal GEOS. Con un semplice accorgimento è pure possibile fare in modo che questi, quando siano terminati, vi riportino nel GEOS. I vostri programmi dovrebbero terminare con la parola chiave END; al posto di questa inserite le linee basic:

1000 print "Inserite il disco di sistema del GEOS." 1010 print "Quando avete terminato premete un 1020 print "tasto qualunque." 1030 get a$:if a$ = "" then 1030 1040 load "geos",8,1

In tal modo ritornerete in GEOS.

GRAFICA E SUONO

per il C64 / C128
C64 Personal Computer

Il Commodore può produrre effetti sonori e grafica fra i migliori ottenibili con un computer. Alcune di queste gradevoli caratteristiche, tuttavia, possono risultare piuttosto complicate da padroneggiare. Anche se non siete degli esperti programmatori, il testo facilita l'apprendimento di quelle tecniche che permettono di apprezzare appieno il proprio calcolatore.

Primi passi
I caratteri grafici - Grafica e POKE - Grafica ad alta risoluzione semplificata.
Formati grafici
La memoria dedicata alla grafica - La grafica "bitmap" - Arte istantanea - Colore di fondo esteso - Come utilizzare formati grafici diversi - Blocco per schizzi ad alta risoluzione
Come ridefinire il set di caratteri
Caratteri personalizzati - Assorbimento completo di caratteri da stampa - Uso evoluto di caratteri grafici
Azione
Costruire gli sprite - Animare gli sprite - Labirinto rotante.
Musica
"Compositore" - "Concertista"
Il sistema operativo GEOS
Comandi fondamentali - GeoPaint - GeoWrite, il word processor - Gli accessori di GEOS
Appendici
Locazioni di memoria - Valori che identificano i colori - Codici ASCII - Codici di schermo - Codici dei tasti - Come usare MLX, programma per scrivere in LM

GRUPPO EDITORIALE JACKSON

L. 35.000 Cod. CC658